Public Service Announcements

- This class will be largely experiential. We will do exercises and debriefs. (I will try—unsuccessfully—to avoid lecture). You’ll do a lot of learning on your own, and with your group.
- The exercises won’t appear in the published proceedings. Take notes!
- I’m a tester. This class intended for everyone, but will be aimed towards testing-related targets from time to time. (Plus: testing and analysis are different, but they live in the same house.)
- I have some ideas, conclusions, and advice, but I’ll have a lot more after the class, because YOU will generate them.
- I can help you to connect ideas to life in the working world, if you can’t see how and if you ask for help. The colleagues around you can help too!
- If you don’t like these ideas, it’s okay with me if you go to someone else’s class! Best to get it over with now.
What Is an Analyst?

- **Analyst**: Anyone for whom analysis is an essential part of their work.
  - programmers
  - technical managers
  - system administrators
  - testers
  - technical support people
  - scientists
  - engineers
  - mechanics
  - researchers
  - educators
  - ...

What Does an Analyst Do?

- Self-directed exploration of systems
  - whether they are physical or abstract; natural or artificial
- Designing and using models of them
  - whether tacit or explicit, formal or informal, exact or approximate
- Designing and applying **heuristics** for that purpose
  - e.g. simple rules and shortcuts, comprehensive procedures, software tools...
- Developing an understanding of those systems

Plus

- Making credible and relevant reports
- Ongoing self-re-education
- Applying critical thinking
Why Analysis for Technical People?

- Development, testing, critical thinking, and analysis are all intertwined.
- Each requires aspects of the other.
- Each helps to reinforce the others.
- They exhibit many similar patterns.
- They require many similar skills.
- They are not linear; they loop.

Analyzing “Analysis”

“decompositional”
“regressive”
“transformative; interpretive”

“up-loosening” (from the literal Greek meaning)
“working back to first principles”
“translation to correct logical form”

breaking down
working backwards
to what is sought

unpacking

de-lumping

drilling down

Stanford Encyclopedia of Philosophy, “Analysis”
https://plato.stanford.edu/entries/analysis
Analysis
To analyze something is to make sense of it; to figure out what it is and what it means.

Synthesis
Because we are trying to make sense of something, analysis also involves synthesis.
Analysis is an exploratory process

Questions About Exploration...

*arrows and cycles*

Where does exploration come from?

What do we do with what we learn?

What happens when the unexpected happens during exploration?

Will everyone explore the same way?

(value seeking)
Questions About Scripted Processes...

*arrows and cycles*

What happens when the unexpected happens during a scripted process?

Where do scripted processes come from?

What do we do with what we learn?

Will everyone follow a scripted process in the same way?

(task performing)

Answers About Scripts...

*arrows and cycles*

What happens when the unexpected happens during a scripted process?

Where do scripted processes come from?

What do we do with what we learn?

Will everyone follow a scripted process in the same way?

(task performing)
Questions About Exploration...

*arrows and cycles*

- Where does exploration come from?
- What do we do with what we learn?
- What happens when the unexpected happens during exploration?
- Will everyone explore the same way?

(value seeking)

Exploration is Not Just Action

*arrows and cycles*

Exploratory behavior acts on itself. Scripted behavior does not.
You can put them together!

arrows and cycles

EXPLORATORY
(value seeking)

SCRIPTED
(task performing)
You can put them together!

_arrows and cycles_

Analysis is a sensemaking process
**Sensemaking**

*Sensemaking in Organizations, by Karl Weick*

1. Grounded in identity construction
2. Retrospective
3. Enactive of sensible environments
4. Social
5. Ongoing
6. Focused on and by extracted cues
7. Driven by plausibility rather than accuracy

---

**Sensemaking for Technical Folk**

*Sensemaking in Organizations, by Karl Weick*

1. Applying our skill sets and mindsets to make sense of...
2. What has happened...
3. As we create conditions, tools, and stories...
4. In collaboration with other people...
5. Continuously throughout the project...
6. To discover what we need to know about the product, and how we might learn about it...
7. Using effective, yet imperfect methods.
Analysis is a heuristic process

Heuristics bring useful structure to problem-solving skill.

- **adjective:**
  
  “serving to discover.”

- **noun:**
  
  “a fallible means of solving a problem.”

A heuristic is not the same as a *rule*.

A heuristic *can* work but *might* fail.

“The engineering method is the use of heuristics to cause the best change in a poorly understood situation within the available resources.”

“The engineer doesn’t know where he is, where he is going, how he will get there, or if anyone will care when he does...The final state always has a reality that the engineer at the initial state cannot anticipate.”

— Billy Vaughan Koen, Discussion of The Method

See “Heuristics for Understanding Heuristics”
http://www.developsense.com/blog/2012/04/heuristics-for-understanding-heuristics/
What is not a heuristic?

A deterministic process that solves a specific problem entirely within a formal system is an algorithm, not a heuristic.

Algorithms give guarantees only within formal systems. Therefore ask: what is this method used for?

Long division is a heuristic when used for equitably dividing a restaurant check.

Fallibility implies that wisdom and responsibility are critical.

Many heuristics are so effective we might be excused for treating them as infallible...

- “Don’t hurt other people.”
- “Breathe now.”
- “Don’t eat things that you find in the street.”

But in engineering, when we treat heuristics as “best practices” we suffer. [note: that is a heuristic, too] For instance...

- “ALWAYS document your work with this template.”
- “ALWAYS write a plan before starting development.”
- “ALWAYS avoid planning before starting development.”
- “ALWAYS use test driven design to write code.”
Why is it dangerous to treat heuristics as “best practices?”

• “Best practices” are simply commandments.
• Commandments are mechanical, uncontrolled, and binary: “do or not do.”
• Anything beyond “just doing it” is therefore illegible, invisible, and seemingly gratuitous.
• Thus, the analyst ordered to use a best practice is given a tacit incentive to use the “commanded heuristic” in the cheapest, laziest way...
• ... and to have no concern for outcome.
• This syndrome is very common. It is called: Pathetic Compliance

Two Ways to Achieve Safety

1. Eliminate freedom
   (Danger: Treats people as objects, requiring expensive vigilance, and inviting rebellion. It also means they have no power to do good things.)

2. Instill responsibility & wisdom
   (Danger: Lecture isn’t enough to do that. Not even this lecture.)
Development of wisdom and responsibility is critical. But how?

- When to use the tool?
- How might the output be distorted?
- What might the output mean?
- What is highlighted? What is suppressed?
- How do you know if it is failing?

Wisdom & Responsibility

Heuristics

Tools that manipulate or transform data (for instance) are heuristics.
The use of that data in decision-making is a heuristic process.

Analysis

...resulting in good decisions.

Responsibility

“I CHOOSE to help (and not harm).”

Wisdom

“I KNOW how to help (and not harm) with my decisions.”

Freedom

“I am ABLE to harm people.” “…or help people.”
Responsibility and wisdom require freedom to develop. Analysis without freedom is nothing but marketing.

“I am **ABLE** to make decisions that may help or harm.”

Managerial (and Parental) Compromise

Curtail freedom only as a **temporary contract** to prevent conflict, or in matters of imminent or extreme danger.

Proceed via **challenges to solve problems**, rather than assignments to perform tasks.

Create lower risk “**sandboxes**” in which high risk activities may be tried.

Cultivate responsibility through **unjustified** respect.

Celebrate the wisdom gained from **mistakes** responsible and free testers (and children) make.
Analysis is a modeling process

Analysis begins and ends with models.

- A model is an idea, activity, or object...
  such as an idea in your mind, a diagram, a list of words, a spreadsheet, a person, a toy, an equation, a demonstration, or a program...
- ...that represents another idea, activity, or object.
  such as something complex that you need to work with or study.
- ...A GOOD model is one that helps you study or understand or manipulate the thing that it represents.
  - A map helps navigate across a terrain.
  - “2 + 2 = 4” is a model for figuring out how many apples are in a basket when we add two apples to a basket that already has two apples in it.
  - Atmospheric models help predict where hurricanes will go.
  - A fashion model helps people to understand how clothing would look on actual humans (okay, really skinny humans).
  - Your beliefs about what you test are a model of what you test.
Some General Systems Analysis Heuristics

- **Cardinality:** Can there be 0, 1, or more than one object?
- **Boundaries:** Is there a limit? More than one? Are different limits consistent?
- **Extrapolation:** If we can go THIS far, can we go FARTHER?
- **Interpolation:** If two things exist in different places, does something exist between them?
- **Intersections:** Do components collide? Can one contaminate another?
- **Surface Integrity:** Does behavior change correctly as input changes in any given dimension?
- **Symmetry/Asymmetry:** If a behavior exists for A, does a corresponding behavior exist for B?
- **Pattern Completion:** Is a pattern apparent that has not yet been completed, or is obscured (all customary parts of a shape)?
- **Negation:** Whatever is there might disappear or reverse.

Some General Systems Analysis Heuristics

- A condition in the world that interacts with your product may happen rarely or frequently.
- A problem may be detectable or undetectable at the time it first occurs or begins to occur (relative to our means of detection).
- A problem may occur in a result, process, component, or environment.
- In the realm of technology and society, problems fall into diverse categories: usability, performance, concurrency, legality, correctness, compatibility, etc.
- A problem may impact a user, a bystander, or the business that produced the product.
- A problem may cause inconvenience, annoyance, or harm.
- ...
Diagrams of Effects (from Weinberg)

• Each node stands for a measurable quantity, like work produced, hours worked, errors created, or errors located.
• Use a “cloud” symbol rather than a circle or rectangle to remind us that nodes indicate measurements, not things or processes as in flow charts, data flow diagrams, etc.
• Cloud nodes may represent actual measurements, or they may represent conceptual measurement
  • things that could be measured, but are not measured just now
  • things that may be too expensive to measure, not worth the trouble, or not measured yet. Key idea: they could be measured (maybe only approximately) if we were willing to pay the price
  • represent actual measurements with a very regular, elliptical “cloud”
  • effects diagrams are mostly for conceptual (rather than mathematical) analysis, so most of the clouds should be appropriately rough

McLuhan’s Laws of Media

• To McLuhan, a medium was anything—a tool, thought or tool that effects some change.
• He proposed “probes”, questions that we can ask about any medium.
  1. What human capability does the new medium extend, enhance, intensify, accelerate, or enable?
  2. What previously obsolete medium does the new medium remind us of (or retrieve)? What effects did that medium have when it arrived?
  3. What (perhaps currently ubiquitous) medium does the new medium obsolesce or make irrelevant?
  4. When extended beyond its original or intended capabilities, how might the new medium reverse its effects?
McLuhan Tetrads: list the effects

- Extends...
- Retrieves...
- Reverses into...
- Obsolesces ...


Excellent analysis both requires and affords critical thinking
The Nature of Critical Thinking

• “Critical thinking is purposeful, self-regulatory judgment which results in interpretation, analysis, evaluation, and inference, as well as explanation of the evidential, conceptual, methodological, criteriological, or contextual considerations upon which that judgment is based.” Critical Thinking: A Statement of Expert Consensus for Purposes of Educational Assessment and Instruction, Dr. Peter Facione

(Critical thinking is, for the most part, about getting all the benefits of your “System 1” thinking reflexes while avoiding self-deception and other mistakes—including overdependence on System 2.)

Bolton’s Definition of Critical Thinking

Critical Thinking
is thinking about thinking
with the aim of not getting fooled.

• Michael Bolton

Testing is enactment of critical thinking about software to help people make better decisions.
Critical thinking must begin with our belief in the likelihood of errors in our thinking.
Reflex is IMPORTANT
But Critical Thinking is About Reflection

System 2

System 1

SEE THINKING FAST AND SLOW, BY DANIEL KAHNEMAN

Levels of Assumptions

<table>
<thead>
<tr>
<th>Reckless</th>
<th>Assumptions that are too risky regardless of how they are managed. Obviously bad assumptions. Don’t make them.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risky</td>
<td>Assumptions that might be wrong or cause trouble, but can be okay with proper management. If you use them, declare them.</td>
</tr>
<tr>
<td>Safe</td>
<td>Assumptions that are acceptable to make without any special management or declaration, but still might cause trouble.</td>
</tr>
<tr>
<td>Required</td>
<td>Assumptions so safe that they cause trouble only IF you manage them, because people will think you are joking, crazy, or insulting.</td>
</tr>
</tbody>
</table>

It is silly to say “don’t make assumptions.” Instead, say “let’s be careful about risky assumptions and avoid the reckless ones.”
Managing Assumptions

Huh? Really? And? So?

Huh?
- You may not understand. (errors in interpreting and modeling a situation, communication errors)

Really?
- What you understand may not be true. (missing information, observations not made, tests not run)

And?
- You may not know the whole story. (perhaps what you see is not all there is)

So?
- The truth may not matter, or may matter much more than you think. (poor understanding of risk)

Looking at something in different ways requires managing distance.
Distance Means Several Things  
the first four definitions from O.E.D.

1. The condition of being at variance; discord, disagreement, dissension; dispute, debate. (original definition in English)
2. Difference, diversity
3. The fact or condition of being apart or far off in space; remoteness
4. The extent of space lying between any two objects; the space to be passed over before reaching an object.

Each of these first three definitions is part of what I am talking about today.

Critical Distance and Social Distance

By critical distance I mean 
A difference between two ways of thinking about some thing, or an absence of knowledge about some thing in favor of other things.

By social distance I mean 
any barrier to or absence of harmony and cooperation among people.

Cultivate critical distance. 
Eliminate social distance.
Deep testing requires critical distance.

Shallow testing doesn’t need critical distance, but deeper or naturalistic long-form testing tends to **require** or **create** more distance from the builder’s mindset.

Why roles? Because changing mindsets is HARD.

- Business analyst skill focus
- Tester skill focus
- Developer skill focus

NOTE: We do NOT claim that different kinds of work **must** be done by different people, or that the people **must** have different titles.

We DO claim that having skilled people **focused** on testing is a powerful heuristic for addressing the mindset switching problem.
Critical Distance can be Helpful

- Critical distance can vastly improve the ability of one process to test, check, or analyze another process.
- It helps prevent shared errors by reducing shared sources of error.
- It helps detect errors that do occur by reducing shared blindness to errors.
- It helps increase innovation, by increasing the variety of our ways of working and increasing the probability of happy accidents.

Social Distance can be Harmful

Social distance can degrade the ability of one process to test or check another process.

- It can cause errors and hide errors by reducing collaboration and sharing of critical information.
- It can reduce motivation for error prevention or detection by reducing empathy and sense of responsibility to outsiders, while increasing defensiveness.
- It can reduce innovation by because of less spontaneous exchange of contrasting ideas.
- It can reduce organizational flexibility and resilience by discouraging different parties from helping each other.
But... Critical Distance and Social Distance are Chained Together

Eliminating social distance also tends to get everyone thinking the same way.

Creating critical distance also tends to reduce the points of connection between people.

Separating them requires ongoing work.

In SOME ways, reducing critical distance can make testing better!

• A highly technical tester who understands the underlying code is automatically going to think MORE like the developer who wrote it.
• This reduces critical distance, which HARMS testing.
• But it also greatly increases his understanding of the product, which HELPS testing.
• My advice: Use open box AND closed box testing!

Although ignorance can sometimes be a good tactic, it is a terrible strategy.
In SOME ways, increasing social distance can make testing better!

- A tester who is remote from and protected from development people does not experience social pressure from them.
- This increases social distance, which HARMs testing.
- But it also greatly reduces coercion and shared motivation to accept a low standard of quality, which HELPS testing.
- My advice: Manage coercion without allowing social disconnection.

This is why managers matter!

Heuristics for Analyzing Claims

- Look for the deep structure of the claim
- Bring suppressed premises and buried assumptions to light
- Negotiate semantics and terms of communication in the trading zone (where people may not share a language)
  - This takes practice! The more you practice the more quickly and easily it happens.
  - Collective tacit knowledge helps! The more you practice as a group, the more quickly and easily it happens.
- This takes some degree of managing
  - your heuristics and biases
  - your feelings about the process
  - your feelings about controversy
A Model for Evaluating Claims

Developed in collaboration with Laurent Bossavit

Deep Dive: Analyzing Risk Analysis
The Product Risk Knowledge Gap:

What we need to know

What we know

Our knowledge of the status of the product.

The purpose of testing is to close the risk gap. The bigger this is, the harder it is to test.

Green is what we’ve already tested or do not need to test.

You can test one thing in one way...

Example: “Let’s look for performance problems across the site.”

Testing is the headlights of the project...

but where should you point them?
Or a different thing in a different way...

Example: “Let’s look for incorrect shipping calculations during checkout.”

Testing is the headlights of the project...
but where should you point them?

You can test shallowly...

Example: “I will see if I can log in.”

Testing is the headlights of the project...
but where should you point them?
Or you can test more deeply...

Example: “I logged in and tested every accessible screen for each of 16 user permission configurations and user account states.”

Testing is the headlights of the project...

but where should you point them?

“Risk-Based Testing” means organizing testing around suspected risk.
How do you know you are in the “green zone?”

You don’t know.

There’s no way to be certain.

We live with that uncertainty.

How do you know decide you are in the “green zone?”

- If you know enough about the product to judge it...
- ...and you feel that its known problems aren’t bad...
- ...and you believe there are (probably) no hidden problems left (that matter)...
- ...and you can explain why you know enough...
- ...and your story is compelling.
**Important Problems?**

**PROBABILITY?**

- Definitely affects everyone continuously
- Affects most people occasionally
- Affects some people occasionally
- Affects few people occasionally
- Affects few people rarely
- Happens only under bizarre conditions

**Important Problems?**

**SEVERITY?**

- Corporate collapse event
- Irreparable damage
- Expensive repairable damage
- Inexpensive repairable damage
- Functionally limiting
- Functionally limiting with workaround
- Merely irritating
- Liveable
**Important Problems?**

**ETHICS and REPUTATION?**

- Shocks public conscience
- Violates law
- Violates standard
- Diminishes or destroys trust
- Makes people wonder “Why didn’t you find that?!?”
- Just unreasonably bad

---

“Risk-Based Testing” means organizing testing around suspected risk.
**Unknown Product Risk**

“Known unknown risk” is when you are **conscious of not knowing** about the risk; you don’t know what to say about the risk other than “this thing might not work.”

Example: “I don’t know if performance is an issue with the online ticket-selling system. I guess I will just have to look at it.”

“Unknown unknown risk” is when **you think you know** where all the high risk is, **but are mistaken**; this leads to overconfidence.

Example: “We run our system on big servers in the cloud. No need for performance testing!”
### Suspected Product Risk

- A risk that you are worried **might** exist regarding the product.
- A risk that you know exists, but **that you don’t yet know enough about**.
- A feeling of concern about some aspect of the product based on generalities or indicators but **without specific, compelling evidence that there is any actual bug in that area**.

Example: “I’m worried about poor performance when people use our ticket-buying system for very popular rock concerts. Peak loads in such cases are astronomical.”

### Known Product Risk

A product risk that you can **describe and explain with sufficient evidence and specificity** that any reasonable person would **accept that it exists**.

Usually, known high risk is based on known open bugs; known low risk is based on good testing that didn’t find bugs.

**Most testing is motivated by the intention to go from speculation about suspected risk to evidence of actual risk.**

Example: “We tested performance under realistic conditions and we now know that it cannot reliably handle more than 150 users per minute. Our servers will crash when Taylor Swift comes to town.”
Actual Product Risk

*Almost all testing results in the assessment of actual risk.*

- A product risk that you can **describe and explain** with **sufficient evidence and specificity** that any reasonable person would **accept that it exists**.
- The presence of any actual bug implies the presence of actual risk.
- We almost never quantify risk in terms of hard probabilities or cold cash because there are usually **too many uncertainties** to model risk in that way.
- In that case, we can use our mental models and our feelings as a heuristic (fast and frugal; letting **plausibility** stand in for accuracy)
- Meanwhile, treat any mathematical or seemingly “complete” and “rational” risk analyses with **great skepticism**.
An Alternative to Calculation

- Risk analysis cannot *really* be about calculation
  - when we talk about risk, we're projecting a problem onto some future
  - problems and bugs can put us into Black Swan domains
  - statistics might influence our ideas about risks
- The alternative to calculation is an attempt to negotiate a set of feelings until we reach a point where we're ready to take action.
  - things get highlighted; things get crystallized
  - leading to vivid potentialities
- Risk analysis is about learning; conjecture; story construction; factoring; quality criteria; systems thinking

Typical Trouble with Product Risk Analysis

- People afraid of anything that has the word “analysis” in it, and looking for easy answers.
- Having no systematic method of risk analysis.
- Conceiving of big huge categories, or tiny specific bugs, but nothing in between.
- Focusing on *project* risks (which are significant because they lead to product risk), but missing the actual product risk
How to Analyze Product Risk

• When you don’t know much
  • You will just be able to get a vague sense of risk.
  • Move toward the product... learn everything you can.
  • Talk to people who know about it.
  • Does the product have a history? Learn about it.
  • Does the product domain have a history? Learn about that.
• Perform survey testing (testing with an emphasis on learning)
• Use circumstantial and general systems heuristics.

The Heuristic Test Strategy Model
https://www.satisfice.com/download/heuristic-test-strategy-model

Oracles
http://www.developsense.com/blog/2012/07/few-hiccups/
How to Analyze Product Risk

- When you know a lot
  - Use the Quality Criteria Categories list from the Heuristic Test Strategy Model to systematically consider different kinds of bugs.
  - Brainstorm risks with the team.
  - List the specific ways the product may fail based on knowledge of the code or knowledge of functionality.
  - List the kinds of problems that users care most about.
  - Consider the problems that have happened before.
  - Use the four-part risk story to fill out your analysis.

The Product Risk Story

“Some person(s) will experience a problem with respect to something desirable that can be detected in some set of conditions because of a vulnerability in the system.”
Risk Story Elements

• Some PERSON(S)
  • user, customer, developer, tester, businessperson, bystander…
  • (a group, a business, a community, society at large…)
• will EXPERIENCE
  • be affected, in the context of an event or situation, at least once by …
• a PROBLEM
  • that leads to bad feelings (annoyance, frustration, confusion), loss, harm, or diminished value…
• with respect to SOMETHING DESIRABLE
  • like capability, reliability, performance…
• that CAN BE DETECTED
  • by a feeling, principle, tool, comparison to a document or picture…
• in SOME SET OF CONDITIONS
  • perhaps always, perhaps only sometimes,…
• because of a VULNERABILITY
  • a bug, a missing feature, an inconsistency…
• in the SYSTEM
  • some result, process, component, feature, environment…

An Example Table of Product Risk Elements

<table>
<thead>
<tr>
<th>Where is the problem?</th>
<th>Can we see it?</th>
<th>When would we see it?</th>
<th>How often does it happen?</th>
<th>How regularly?</th>
<th>What is affected?</th>
<th>Quality criterion?</th>
<th>Who is affected?</th>
<th>How do they feel?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure</td>
<td>Obvious</td>
<td>Immediately</td>
<td>Consistent</td>
<td>Structure</td>
<td>Capability</td>
<td>Society</td>
<td>Impatient</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Obscure</td>
<td>Later</td>
<td>Rarely</td>
<td>Intermittent</td>
<td>Function</td>
<td>Public</td>
<td>Confused</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>Invisible</td>
<td>Never</td>
<td>Data</td>
<td>Interfaces</td>
<td>Usability</td>
<td>Consumer</td>
<td>Annoyed</td>
<td></td>
</tr>
<tr>
<td>Interfaces</td>
<td>Can others?</td>
<td></td>
<td></td>
<td>Interfaces</td>
<td>Charisma</td>
<td>Network Admin</td>
<td>Surprised</td>
<td></td>
</tr>
<tr>
<td>Platform</td>
<td></td>
<td></td>
<td></td>
<td>Upstream</td>
<td>Security</td>
<td>External Developer</td>
<td>Disappointed</td>
<td></td>
</tr>
<tr>
<td>Operations</td>
<td></td>
<td></td>
<td></td>
<td>Downstream</td>
<td>Scalability</td>
<td>Internal Developer</td>
<td>Angry</td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td>Coexistence</td>
<td>Tester</td>
<td>Afraid</td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
<td>Time</td>
<td>Interoperability</td>
<td>Tech writer</td>
<td>Suspicious</td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10-Risk - 7
Analyzing Risk

• Consider creating and maintaining a **product** risk list and a **project** risk list, especially if no one else on the team is doing it.

• Brainstorm a list of risks, and rank them in order of significance.

• Then compare this list to coverage and quality criteria areas in the Heuristic Test Strategy Model, or in your own taxonomies.

• Identify tasks associated with investigating and managing risks. Make it public, and advertise when and where you need help.

• Do some testing *not* focused on specific risks, in order to discover unrecognized risks.

See RST Appendices for several examples, including “Install Risk Catalog”, “Risk-Based Test Plan (OWL)”, “Risk-Based Test Plan #2”.

Not Everything Needs Deep Testing

• Only build easy things.

• Stop caring if it’s broken.

• Build in a testable way.
Activities in Analysis

- Identifying goals
- Modeling the analysis space
- Collecting observations and data
- Evaluating the quality of the data
- Identify factors that influence evaluations
- Identify factors that influence each other
- Classifying and coding (in the qualitative research sense) data
- Comparing and contrasting
- Seeking patterns and connections
- Identifying consistencies and inconsistencies
- Accounting for inconsistencies
- Developing explanations
- Managing assumptions and validity

Managing Your Data

- Gathering it
- Arranging it
- Presenting it
- Representing it
- Organizing it
- Storing it
- Processing it
Analysis Meta-Methods

- Developing descriptions
  - narrating, writing
  - seeing what’s being said and noticing what’s being left out
- Visualizing the data and relationships
  - drawing, sketching
  - arranging maps, tables, charts, schematics...
  - focusing; defocusing; refocusing
  - seeing what’s there and noticing what’s missing
- Managing validity
- Reflecting
- Revising

Analysis Methods: Examples of Alternation

- Forward Analysis ⇔ Backward Analysis
  - Open Coding vs. Closed Coding
  - Forward Chaining vs. Backward Chaining Reasoning
- Incubating ⇔ Progressing
- Focusing ⇔ Defocusing
- Touring ⇔ Sensemaking
- Lone ⇔ Social
- Talking ⇔ Doing
- Planning ⇔ Doing
- Suppression ⇔ Exaggeration
- Naturalistic ⇔ Artificial ⇔ Pathological
- Easy ⇔ Hard
Analysis Method: Scaling and Scoping

- Examine an object, phenomenon, or idea at different scales.
- Look at the whole.
- “Reduce it” to its elements.
- Reduce its elements to their elements.
- Examine the relationship of the object to things around it.
  - What are those things?
  - What are the relationships?
- What is missing or invisible?
- What changes when you change the scale at which you’re looking?
- Observe how things change over time.

Problems in Analysis

Any analysis is vulnerable to problems by attending to some factors and ignoring or ruling out others in...

- Conceptual frameworks
- Research questions
- Sampling; examination of instances
- Instrumentation
- The nature and quality of your data

Workaround: apply critical thinking.
Readings

- Stanford Encyclopedia of Philosophy, “Analysis”
- https://plato.stanford.edu/entries/analysis/
- Elkins, James. How to Use Your Eyes.

Readings

Readings


A Word from Our Sponsor (Me)

- Rapid Software Testing is a course, a mind-set, and a skill set about how to do excellent software testing in a way that is very fast, inexpensive, credible, and accountable. I co-author RST with James Bach.
- I teach RST in classes for testers, developers, managers, business analysts, documenters, DevOps people, tech support...
- I also offer advice and consulting on testing and development to those people, and also to managers and executives.

http://www.developsense.com