

Requirements 2, Interviews, and Risk

Claire Le Goues

Michael Hilton

Learning goals

- Define and identify stakeholders.
- Demonstrate basic proficiency in executing effective requirements interviews.
- Evaluate risk for a product

Administrivia

- Homework 2
 - Initial planning document and backlog (DUE today)
 - Technical Artifacts (DUE 9/29)
 - Reflection Documents (DUE 10/1)

Josh Gardner

Interview Follow-up

- Observations?
 - Anything surprising? Unexpected?
 - Confirmations of existing ideas?
 - Generalizable knowledge?

Interviews

REQUIREMENTS ELICITATION

Typical Steps

- Identify stakeholders
- Understand the domain
 - Analyze artifacts, interact with stakeholders
- Discover the real needs
 - Interview stakeholders
- Explore alternatives to address needs

Question

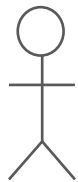
- Who is the system for?
- Stakeholders:
 - End users
 - System administrators
 - Engineers maintaining the system
 - Business managers
 - ...who else?

Stakeholder

- Any person or group who will be affected by the system, directly or indirectly.
- Stakeholders may disagree.
- Requirements process should trigger negotiation to resolve conflicts.

Defining actors/agents

- An actor is an entity that interacts with the system for the purpose of completing an event [Jacobson, 1992].
 - Not as broad as stakeholders.
- Actors can be a user, an organization, a device, or an external system.



Sales
Specialist



Marketing



GPS
Receiver



Inventory
System

Stakeholder analysis: criteria for identifying relevant stakeholders

- Relevant positions in the organization
- Effective role in making decisions about the system
- Level of domain expertise
- Exposure to perceived problems
- Influence in system acceptance
- Personal objectives and conflicts of interest

Stakeholders, a NASA example



From HSI NAP 11893

FIGURE 6-3 Role network for National Aeronautics and Space Administration (NASA's) Near Earth Asteroid Rendezvous project.

Challenges

- Distributed knowledge
- Conflicting knowledge
- Difficult to access sources
- Communication barriers (cultural, terminology, backgrounds)
- Hidden needs, tacit knowledge
- Politics, unstable conditions

Studying Artifacts (Content Analysis)

- Learn about the domain
 - Books, articles, Wikipedia
- Learn about the system to be replaced
 - How does it work? What are the problems? Manuals? Bug reports?
- Learn about the organization
- Knowledge reuse from other systems?

Interviews



Interview Tradeoffs

- Strengths
 - What stakeholders do, feel, prefer
 - How they interact with the system
 - Challenges with current systems
- Weaknesses
 - Subjective, inconsistencies
 - Capturing domain knowledge
 - Familiarity
 - Technical subtlety
 - Organizational issues, such as politics
 - Hinges on interviewer skill

Interview Process

- Identify stakeholder of interest and target information to be gathered.
- Conduct interview.
 - (structured/unstructured, individual/group)
- Record + transcribe interview
- Report important findings.
- Check validity of report with interviewee.

Example: Identifying Problems

- What problems do you run into in your day-to-day work? Is there a standard way of solving it, or do you have a workaround?
 - Why is this a problem? How do you solve the problem today? How would you ideally like to solve the problem?
- Keep asking follow-up questions (“What else is a problem for you?”, “Are there other things that give you trouble?”) for as long as the interviewee has more problems to describe.
- So, as I understand it, you are experiencing the following problems/needs (describe the interviewee’s problems and needs in your own words – often you will discover that you do not share the same image. It is very very common to not understand each other even if at first you think you do).
- Just to confirm, have I correctly understood the problems you have with the current solution?
- Are there any other problems you’re experiencing? If so, what are they?

Capturing v. Synthesizing

- Engineers acquire requirements from many sources
 - Elicit from stakeholders
 - Extract from policies or other documentation
 - Synthesize from above + estimation and invention
- Because stakeholders do not always know what they want, engineers must...
 - Be faithful to stakeholder needs and expectations
 - Anticipate additional needs and risks
 - Validate that “additional needs” are necessary or desired

Interview Advice

- Get basic facts about the interviewee before (role, responsibilities, ...)
- Review interview questions before interview
- Begin concretely with specific questions, proposals; work through prototype or scenario
 - Relate to current system, if applicable.
- Be open-minded; explore additional issues that arise naturally, but stay focused on the system.
- Contrast with current system/alternatives. Explore conflicts and priorities
- Plan for follow-up questions

Bonus: Guidelines for effective interviews

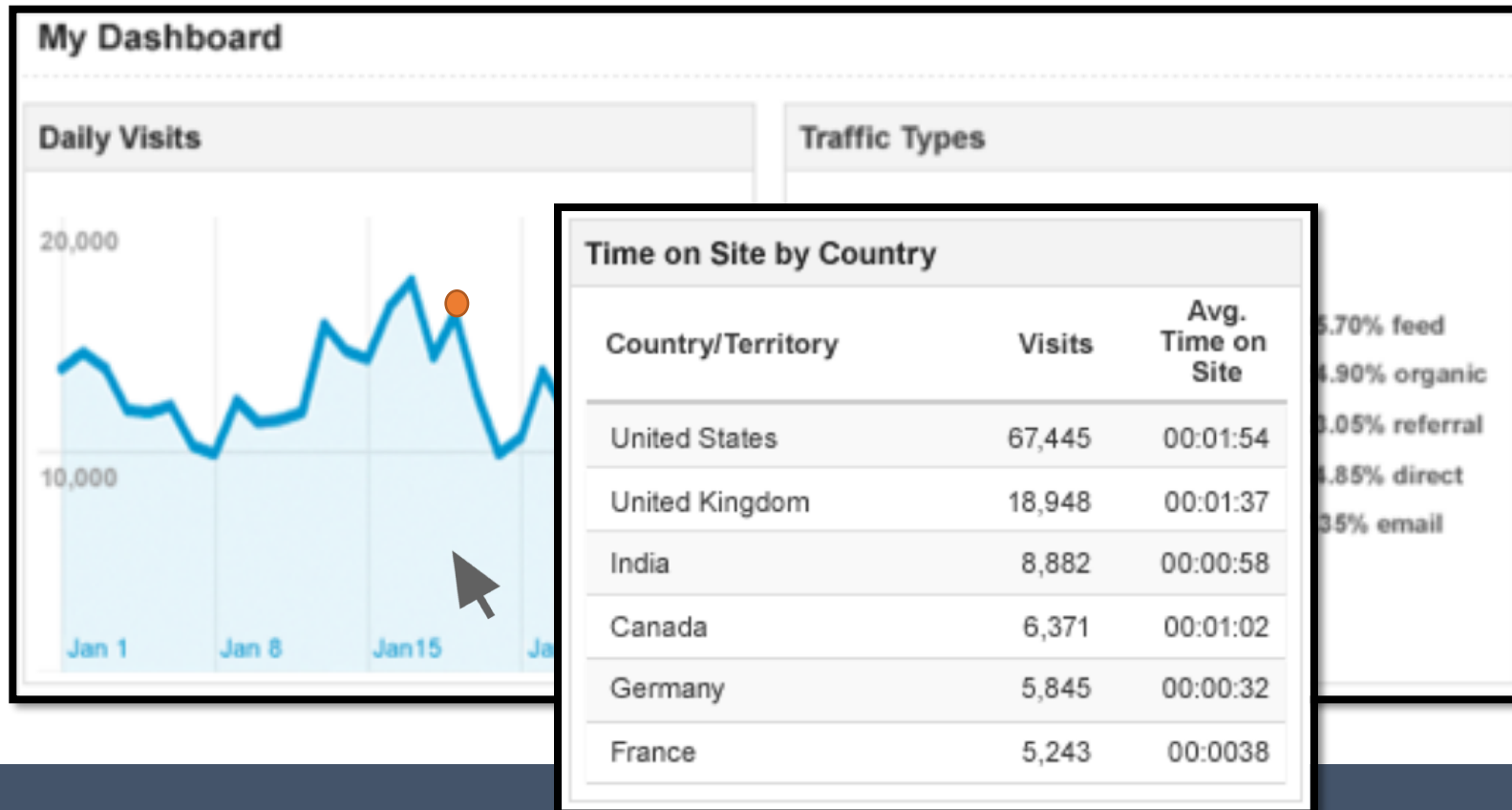
- Identify the right interviewee sample for full coverage of issues
 - different responsibilities, expertise, tasks, exposure to problems
- Come prepared, to focus on right issue at right time
 - background study first
 - predesign a sequence of questions for this interviewee
- Centre the interview on the interviewee's work & concerns
- Keep control over the interview
- Make the interviewee feel comfortable
 - Start: break ice, provide motivation, ask easy questions
 - Consider the person too, not only the role
 - Do always appear as a trustworthy partner

Bonus: Guidelines for effective interviews

- Be focused, keep open-ended questions for the end
- Be open-minded, flexible in case of unexpected answers
- Ask why-questions without being offending
- Avoid certain types of questions ...
 - opinion or biased
 - affirmative
 - obvious or impossible answer for this interviewee
- Edit & structure interview transcripts while still fresh in mind
 - including personal reactions, attitudes, etc
- Keep interviewee in the loop
 - co-review interview transcript for validation & refinement

PROTOTYPES, MOCKUPS, STORIES

High- vs low- fidelity mockups



Mockups, Prototypes, Stories

- Humans: better at recognizing whether a solution is correct than solving the problem from a blank page.
- Mock-ups/prototypes help explore uncertainty in the requirements.
 - Validate that we have the right requirements.
 - Elicit requirements at the “borders” of the system.
 - Assert feasibility of solution space.
 - Get feedback on a candidate solution.
- “I’ll know it when I see it”

Rapid prototyping

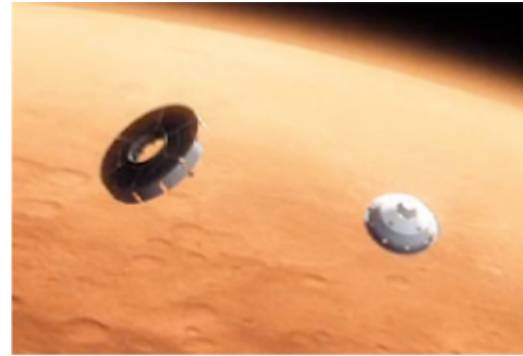
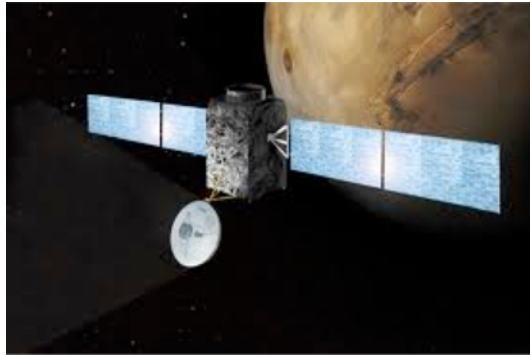
- Throw-away: developed to learn more about a problem, not intended for actual use.

-



- Evolutionary: intended to be incorporated into the final product.

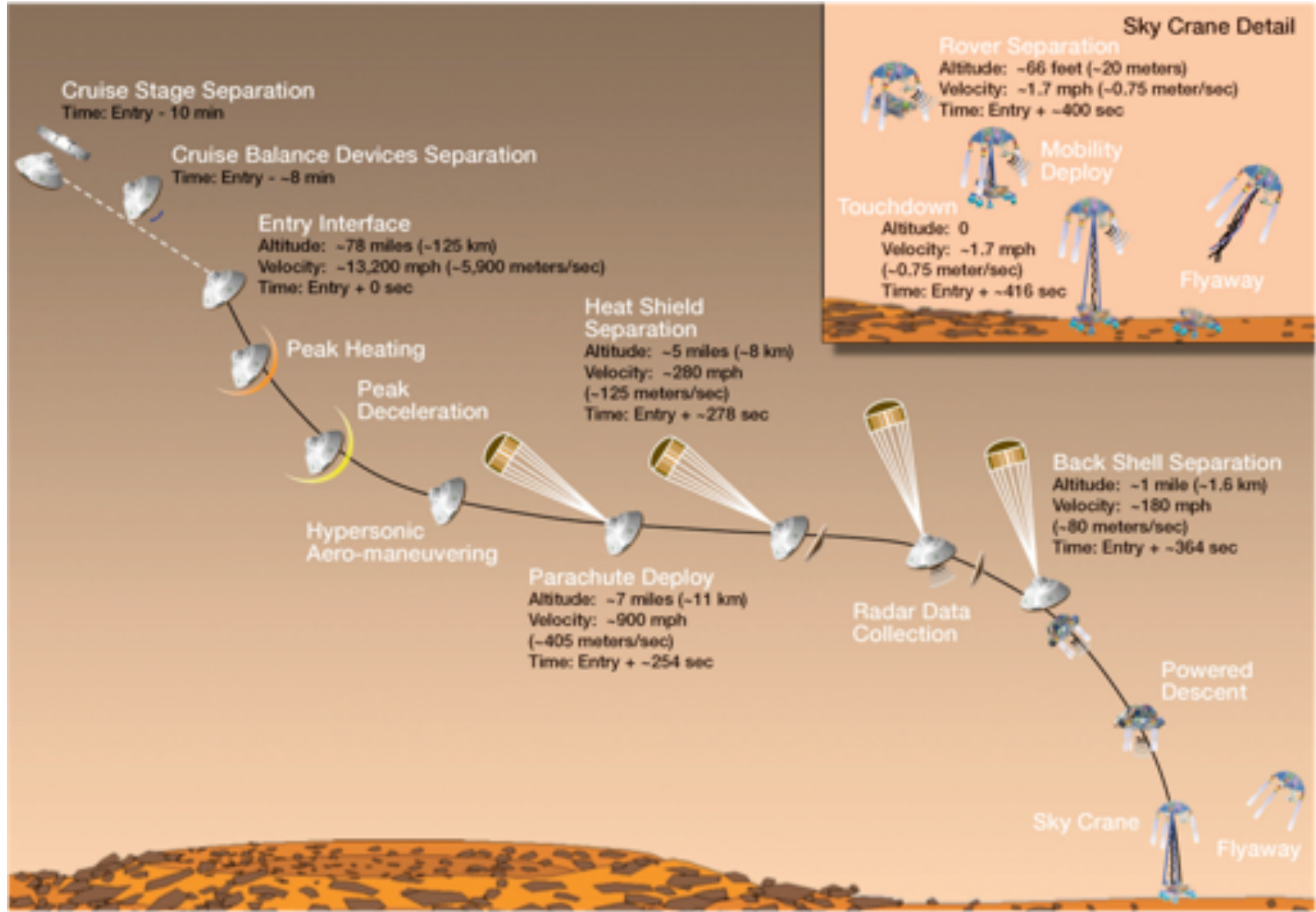
Storyboarding and scenarios



Story

- **Who** the players are
- **What** happens to them
- **How** it happens through specific episode
- **Why** this happens
- **What** if such and such an event occurs
- **What** could go wrong as a consequence

- **Storyboards illustrate scenarios:** a typical sequence of interaction among system components that meets an implicit objective.
 - **Storyboards** explicitly cover at least **who, what, and how**.
- Different types:
 - Positive vs negative (should and should not happen)
 - Normal vs abnormal
- As part of elicitation:
 - Learn about current or proposed system by walking through real-life or hypothetical sequences
 - Can ask specific questions
 - Elicit the underlying objectives, generalize into models of desired behaviors.
 - Identify and resolve conflicts
- Pluses: Concrete, support narrative description
- Minuses: inherently partial.



RESOLVING CONFLICTS

Types of inconsistency

- Terminology clash: same concept named differently in different statements
 - e.g. library management: “borrower” vs. “patron”
- Designation clash: same name for different concepts in different statements
 - e.g. “user” for “library user” vs. “library software user”
- Structure clash: same concept structured differently in different statements
 - e.g. “latest return date” as time point (e.g. Fri 5pm)
 - vs. time interval (e.g. Friday)

Types of inconsistency, 2

- Strong conflict: statements not satisfiable together
 - e.g. “participant constraints may not be disclosed to anyone else” vs. “the meeting initiator should know participant constraints”
- Weak conflict (divergence): statements not satisfiable together under some boundary condition
 - “patrons shall return borrowed copies within X weeks” vs “patrons shall keep borrowed copies as long as needed” contradict only if “needed > x weeks”

Handling inconsistencies

- Terminology, designation, structure: Build glossary
- Weak, strong conflicts: Negotiation required
 - Cause: different objectives of stakeholders => resolve outside of requirements
 - Cause: quality tradeoffs => explore preferences

Requirements Traceability

- Keep connections between requirements
- What follows from what

Requirements prioritization

- Cost, time, and other limits
- Dependencies among requirements
- Nice to have

- Strategies to base on value contribution

Summary

- Many solicitation strategies, including document analysis, interviews, and ethnography
- Do not underestimate the challenge of interviews
- Resolving conflicts
- Using prototypes to enhance discussions and decision making
- Many documentation strategies; our focus is on *user stories*

Risk

Risk

 **Tony Webster** 
@webster Follow 

I appreciate the honesty.

Pick a password

Don't reuse your bank password, we didn't spend a lot on security for this app.
At least 6 characters

Continue

8:20 PM - 15 Sep 2018

5,868 Retweets 15,672 Likes 

 58  5.9K  16K 

What are risks?

- A **risk** is an uncertain factor that may result in a loss of satisfaction of a corresponding objective

For example...

- System delivers a radiation overdose to patients (Therac-25, Theratron-780)
- Medication administration record (MAR) knockout
- Premier Election Solutions vote-dropping “glitch”

How to assess the level of risk?

- Risks consist of multiple parts:
 - Likelihood of failure
 - Negative consequences or impact of failure
 - Causal agent and weakness (in advanced models)
- Risk = Likelihood x Impact

CVSS V2.10 Scoring

The Common Vulnerability Scoring System consists of:

- 6 base metrics (access vector, complexity, confidentiality impact, ...)
- 3 temporal metrics (exploitability, remediation, ...)
- 5 environmental metrics; all qualitative ratings (collateral damage, ...)

BaseScore =

$\text{round_to_1_decimal}(((0.6 * \text{Impact}) + (0.4 * \text{Exploitability}) - 1.5) * f(\text{Impact}))$

Impact =

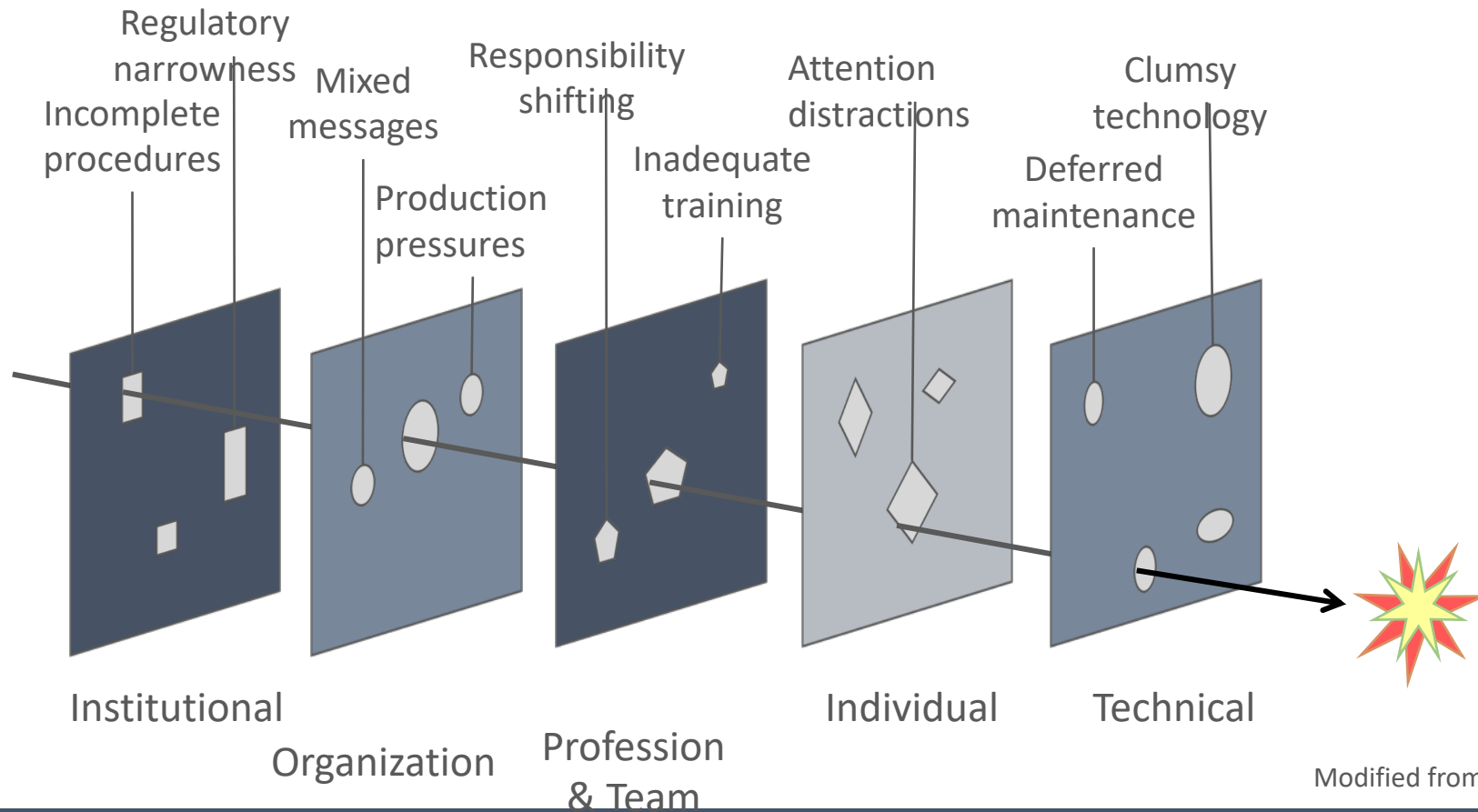
$10.41 * (1 - (1 - \text{ConfImpact}) * (1 - \text{IntegImpact}) * (1 - \text{AvailImpact}))$

Exploitability =

$20 * \text{AccessVector} * \text{AccessComplexity} * \text{Authentication}$

f(impact) = 0 if Impact=0, 1.176 otherwise

The Swiss cheese model



Modified from Reason, 1999, by R.I. Crook

Aviation failure impact categories

- **No effect** – failure has no impact on safety, aircraft operation, or crew workload
- **Minor** – failure is noticeable, causing passenger inconvenience or flight plan change
- **Major** – failure is significant, causing passenger discomfort and slight workload increase
- **Hazardous** – high workload, serious or fatal injuries
- **Catastrophic** – loss of critical function to safely fly and land

Risk assessment matrix



- MIL-STD-882E

<https://www.system-safety.org/Documents/MIL-STD-882E.pdf>

TABLE III. Risk assessment matrix

RISK ASSESSMENT MATRIX				
SEVERITY PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			



DECIDE Model

Detect that the action necessary

Estimate the significance of the action

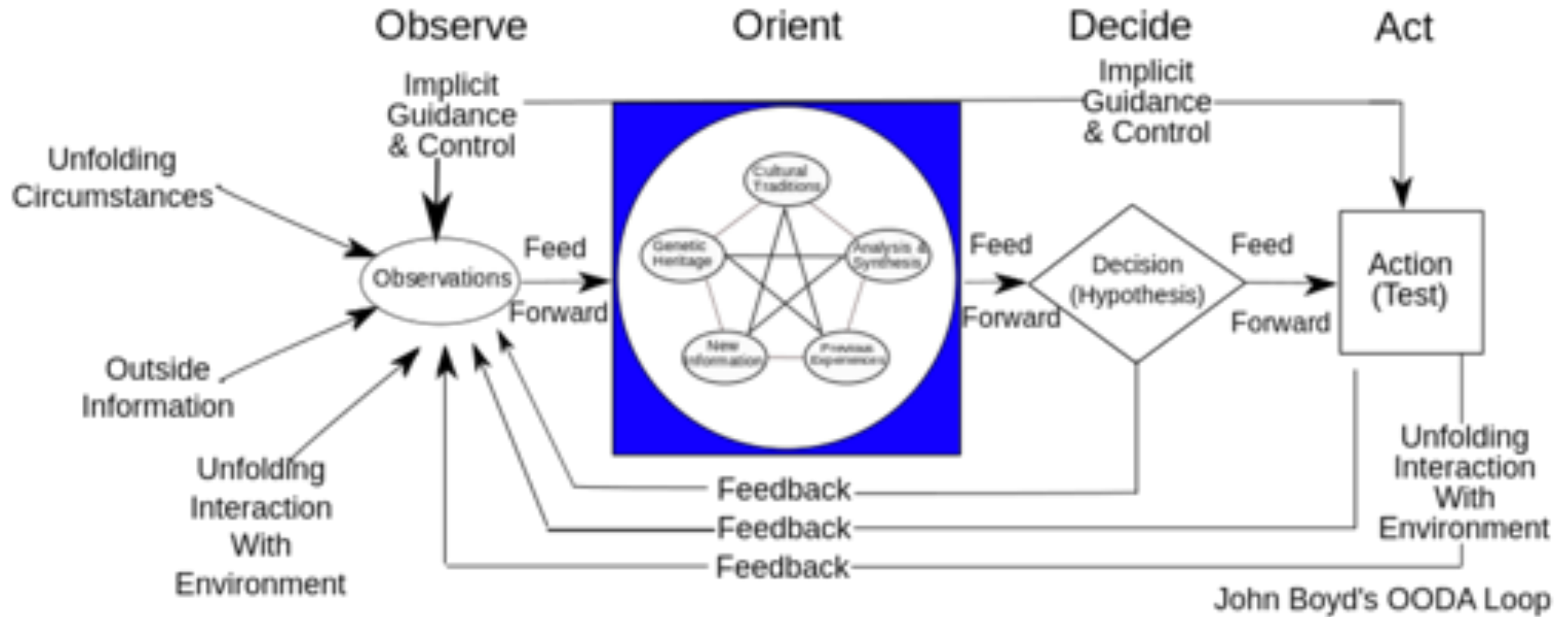
Choose a desirable outcome

Identify actions needed in order to achieve the chosen option

Do the necessary action to achieve change

Evaluate the effects of the action

OODA Loop



Exercise



Bird Risks



Further Reading

- Larman, Craig. *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development*. Pearson, 2012. Chap. 6
- Van Lamsweerde A. *Requirements engineering: From system goals to UML models to software*. John Wiley & Sons; 2009. Chapter 2-4