Lecture 4

Wrapping ambiguity

Starting your project

Agenda

Guest Presentation:

Aadeel Akhtar

Review:

Ambiguity

Lecture:

Project Introductions

Guest Lecture

- Aadeel Akhtar

- Stating the obvious….

> “If you do not know what you want, you are quite unlikely to get it.”

- In engineering design, the description of what you want is captured by the problem statement and “high-level” requirements.
- We call them “high-level” to differentiate them from lower level specifications.

Example:

- Requirement - The car must be safer than the current model.
- Specification - The airbags must deploy in under 0.3 seconds.

Example from: “Exploring requirements – quality before design” – V

Ambiguity - the source of all that is evil…..

- Missing requirements
  - Examples?
- Ambiguous language
  - Words
    - Small, large, lots, many, very, group
- Introduced elements
  - Notice our definition never actually said structure…we made that assumption ourselves.

Example from: “Exploring requirements – quality before design” – V

Answers from Gause and Weinberg (100 people at conference)

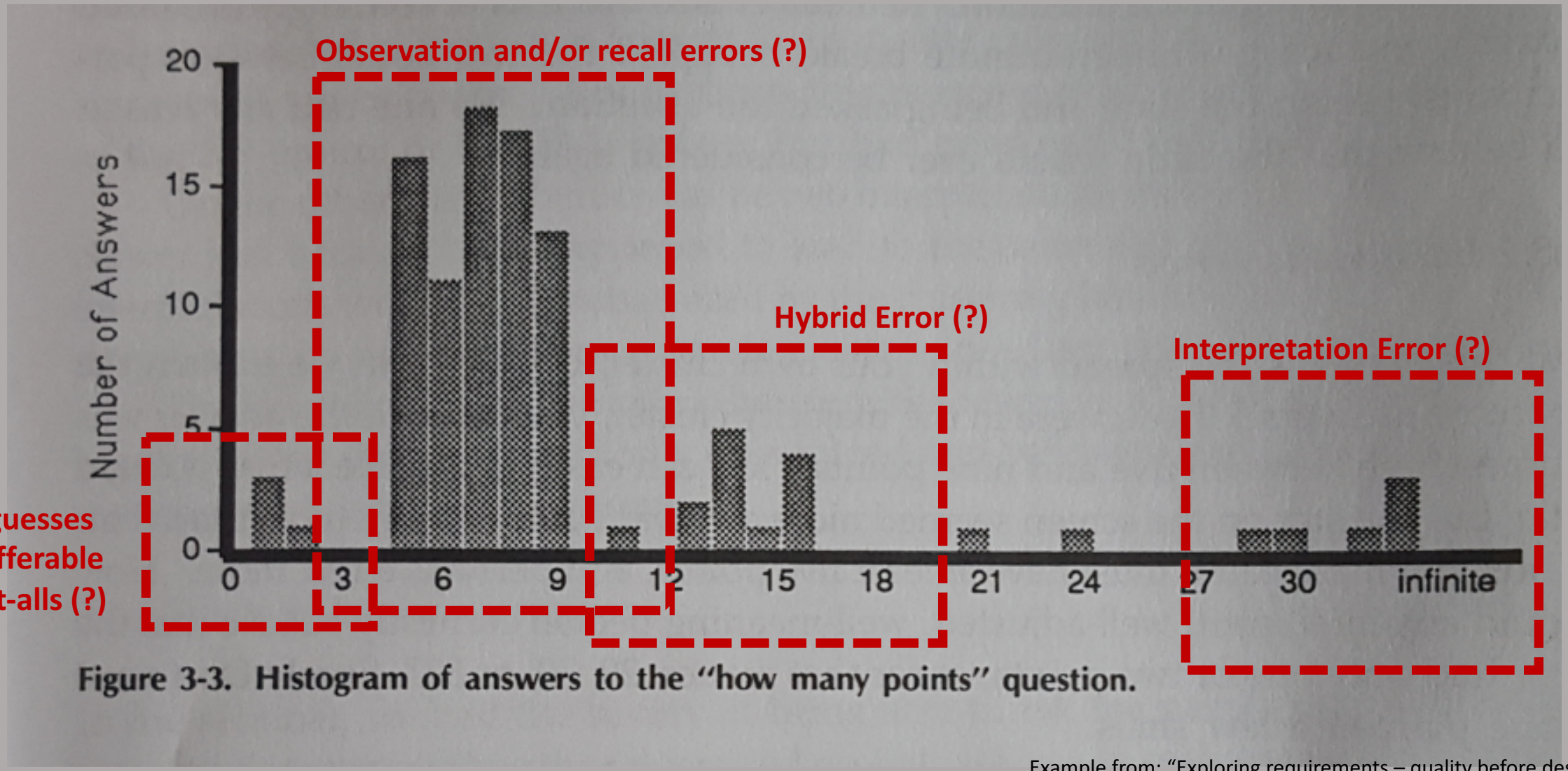


Figure 3-3. Histogram of answers to the "how many points" question.

Example from: "Exploring requirements – quality before design" – V

Methods for Combating Ambiguity?

- Open discussion
  - Was our class answer better than the individual answers?
  - Why?
- Direct questions
  - We can envision our design space as a decision tree, the problem statement as the root of that decision tree, and the requirements as a set of decisions to help us navigate that tree.
  - By asking direct questions, we can intuitively navigate that tree.
- Starting points
- Context free questions
- Considering the customer and user
- Brainstorming

Example from: “Exploring requirements – quality before design” – V

Context Free Questions

- A set of questions that can be asked regardless of the design project.

- Context Free Process Questions
  - Who is the client for the XXX project?
  - What is a highly successful solution really worth to this client?
  - What is the real reason for wanting to solve this problem?
  - Should we use a single team? More than one? Who should be on those teams?
  - How much time do we have?
  - Where else can the solution to this problem be obtained? Can we copy an existing solution?

Example from: “Exploring requirements – quality before design” – V

Context Free Questions

- A set of questions that can be asked regardless of the design project.

- Context Free Process Questions
  - Who is the client for the XXX project?
  - What is a highly successful solution really worth to this client?
  - What is the real reason for wanting to solve this problem?
  - Should we use a single team? More than one? Who should be on those teams?
  - How much time do we have?
  - Where else can the solution to this problem be obtained? Can we copy an existing solution?

  - Cowboy example

Example from: “Exploring requirements – quality before design” – V

Context Free Product Questions

- What problems does this system solve?
- What problems could this system create?
- What environment is this system likely to encounter?
- What kind of precision is required?

Example from: “Exploring requirements – quality before design” – V

Metaquestions – questions about questions

- Do my questions seem relevant?
- Are you the right person to answer this question?
- May I write your answers down and read them back to you?
- Is there someplace we can see the environment where this product will be used?
- Is there anything else I should be asking you?
- Is there anything you would like to ask me?
- May I ask you more questions later?

Example from: “Exploring requirements – quality before design” – V

Advantages

- Can have a list of questions to answer in advance.
- They can help with social awkwardness if working with an external customer.
- Help you focus on high level issues.
- Help raise issues that may have been implicit.
- Can help to elucidate competing assumptions.

Getting the right people involved

- Often, and very often in 445, we design something without the MOST important people in mind….the customers and the users.
  - Customer (or client)
    - The people pay you for your product
    - "Who is the customer" should always be your first context-free question.
  - User - anyone who is affected by the project
    - Customers are users…
    - Users are not always the customers
      - Why wouldn't a line of toys that children love sell?

Why include the user?

- Customers pay for the product
- Users make or break products, many tools are never used, without use…there will be no repeat customers…
- Users should be involved in the design process as well

The railroad paradox

1. Product is not satisfactory.
2. Because of 1, no one uses the product.
3. Potential users ask for a better product.
4. Because of 2, request is denied.

Products can create new users…

- Speed limits and traffic fatalities

Optimizing a product

- May be possible to improve a product for some without making it worse for others…Pareto optimization.
- Others (Veblen) argue that making a product better for one user always makes it worse for some others.
- When we change something in a design, we should consider both groups.

Considering important users

- Identifying users and whether they should be dealt with is important.
- Start with a huge list
- Prune list
- Then, for a design we can consider whether the it should be friendly to them, ignore them, or be unfriendly to them
  - Unfriendly example, medicine bottles for children

Participation

- Who should be involved?
  - Ideally, every potential user
  - Sometimes this is possible
  - Often is not…if not, take a sample?
  - Don't confuse a sample with being "everyone".
- When should they be involved?
  - Sometimes they are members of the team.
    - Can be useful if there is a small user group known ahead of time.
  - Part time user involvement more common
    - Have a plan…may drift to 0.

Example from: "Exploring requirements – quality before design" – V

Participation

- How to get judgements?
  - Mockups
  - Models
  - Be careful to differentiate from opinions and data from experiments
  - Make note of whether your representations of the product are representative of the real world.
    - Example: tops of buildings…

- Final note: when it comes to including customers and users, have a plan and make it public.

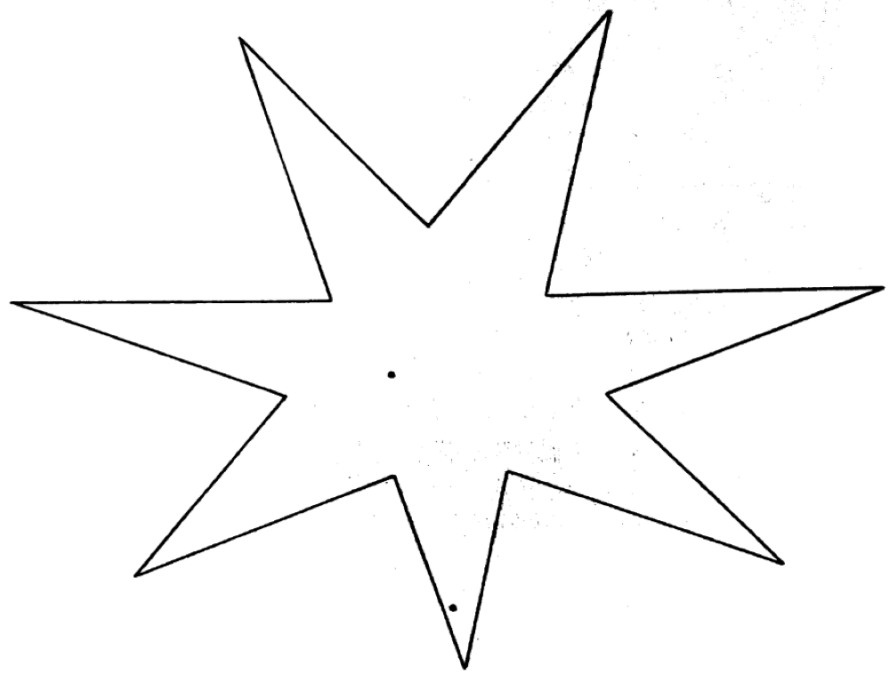
Participation

- How to get judgements?
  - Mockups
  - Models
  - Be careful to differentiate from opinions and data from experiments
  - Make note of whether your representations of the product are representative of the real world.
    - Example: tops of buildings…
- Final note: when it comes to including customers and users, have a plan and make it public.

Example from: “Exploring requirements – quality before design” – V

Ambiguity Reduction Heuristics

- Memorization
  - Note: According to Gause and Weinberg…programmers can find bugs easier if they try to memorize code. Areas that are harder to memorize more likely to be in error.
  - If a problem statement or requirement are clear, they may be easier to remember.
  - Try having the team memorize them, analyze errors to determine what is unclear.



- Poll people
  - How many points were in the star?
  - After time and discussion re-poll.

Example from: “Exploring requirements – quality before design” – V

Mary had a little lamb

- Analyze things word for word and then in combinations of words
  - Meaning may not be clear or can be lost over time
    - Half a pound of tupenny rice,
Half a pound of treacle.
That's the way the money goes,
Pop! goes the weasel.
  - What does "Mary had a little lamb" tell us?
    - *Mary* had a little lamb
    - Mary *had* a little lamb
    - *Mary had* a little lamb

Example from: "Exploring requirements – quality before design" – V

Mary conned the trader

- What if we were to substitute synonyms for the words in the problem statement?

"Mary had a little lamb"

- Had
  - Possess own hold
  - To obtain, accept
  - To partake
  - Trick
- Lamb
  - Young sheep, less than 1 year of age
  - Young of various animals
  - Gentle or weak person
  - Easily cheated or deceived, especially in securities

Examples

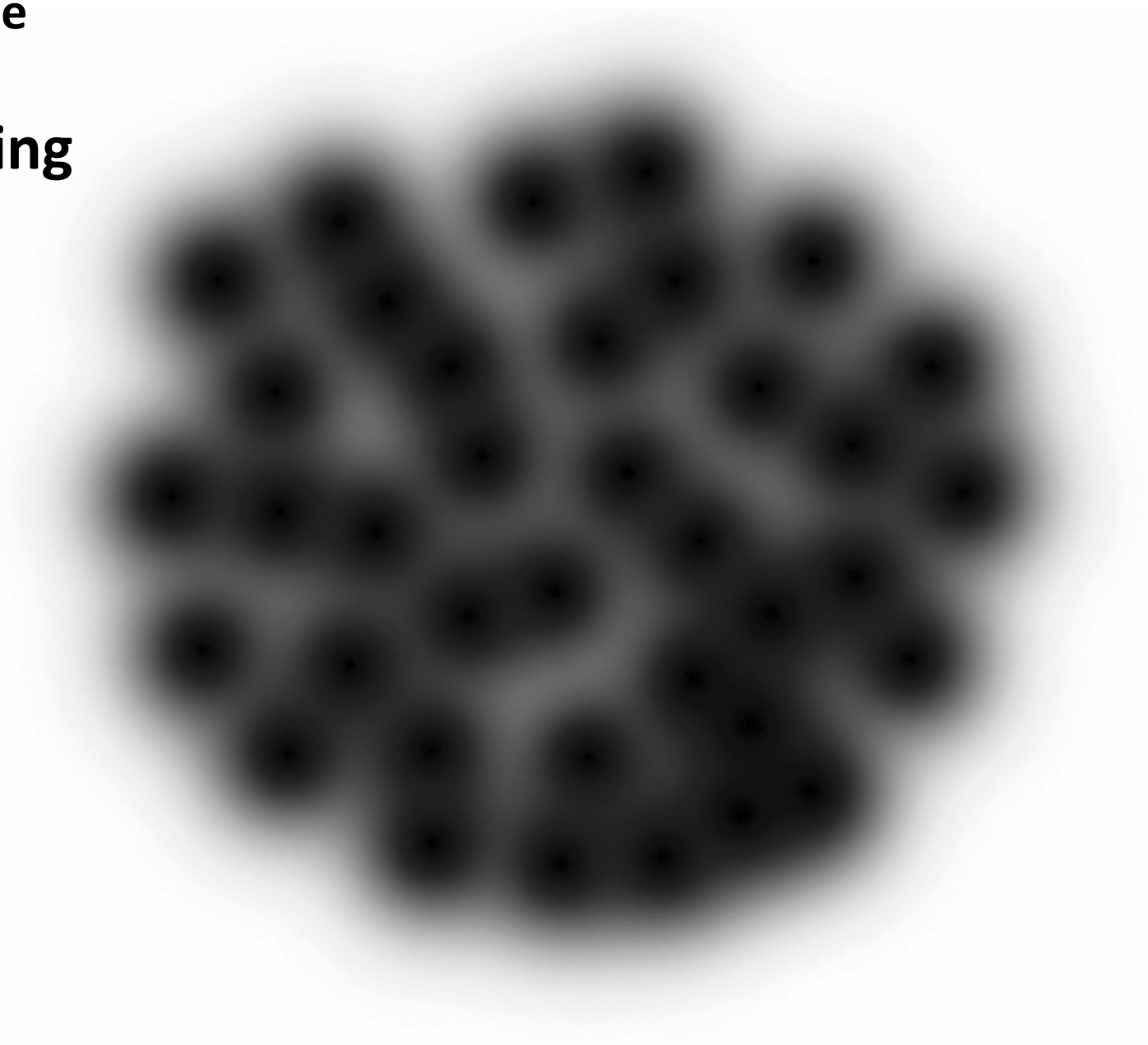
- Mary owned a young antelope
- Mary had a little bit of lamp for dinner
- Mary tricked a little sheep under one year of age
- Mary had a small gentle person
- Mary bribed a small person trading in securities who was easily cheated

Example from: "Exploring requirements – quality before design" – V

A problem statement and a set of requirements

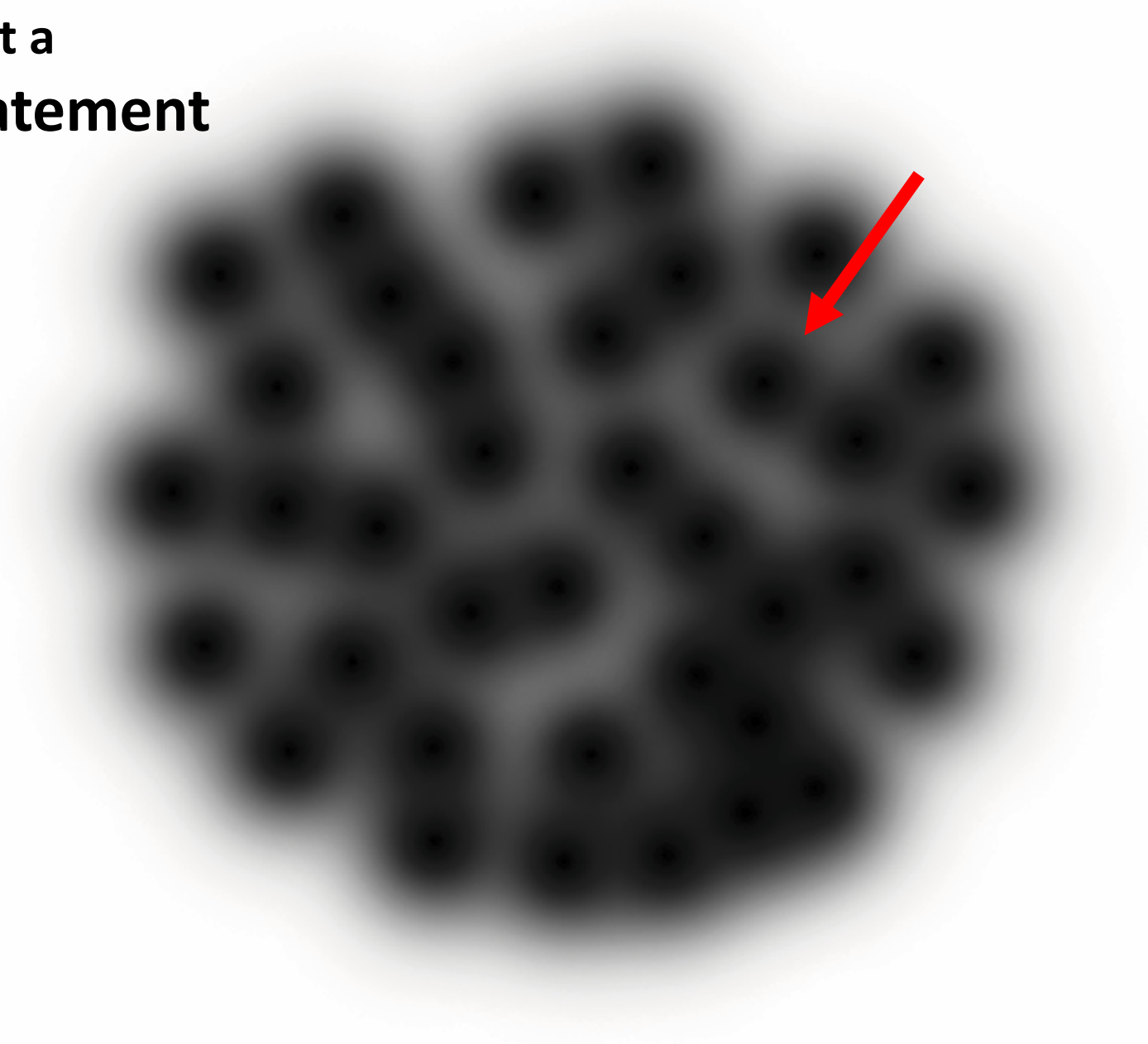
Our initial conception
of a
Project Space

Project Space
after
Brainstorming



Use idea reduction to arrive at a

Problem Statement



From the Problem Statement we can see that there are multiple things we must do to solve the problem

We have been referring to these sub-problems as our

“High-Level” Requirements

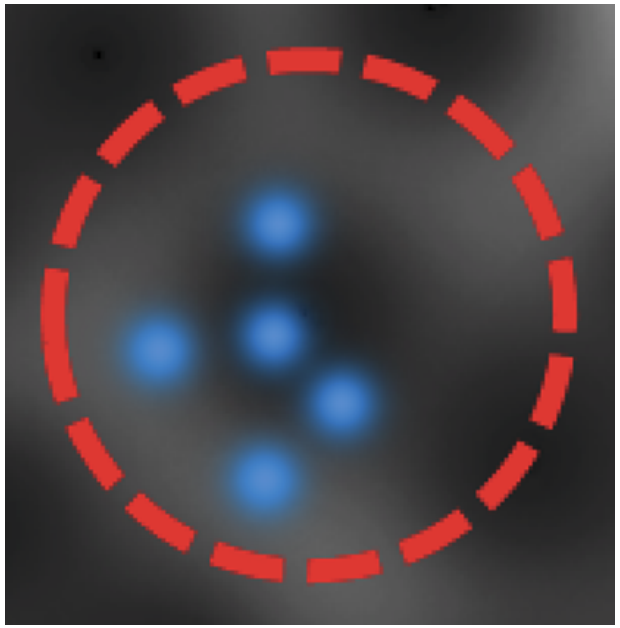
Combination of

Clear Problem Statement

and

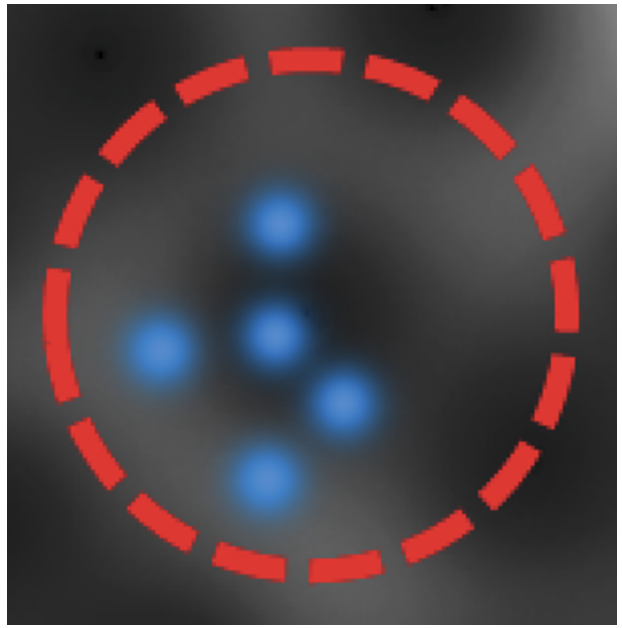
Unambiguous Requirements

give us our victory conditions



These victory conditions can be thought of as bounds on our design, otherwise known as a

Problem Space



One more source of ambiguity…the project's name.

Examples from the 445 project history:

Smart Window Responding System

Smart Hiking Bladder

Smart Home Temperature System with Fancy Spinning LED Display

Smart Lock With Smart Keys

One more source of ambiguity…the project's name.

- Time permitting, activity.

Influence of a name

- Time permitting, activity.

One more source of ambiguity…the project's name.

- How would rate your group's dynamic?
- How many people from each group asked to present?
- Clever solutions as judged by others?

The Project's Name

- A name is a project's public face, it may be the only thing someone knows about your project.
- Names are used repeatedly and tend to steer the way we think about solutions.

- The name of a project can influence perception and behavior within a group, even in the case when we have no way of comparing the name with other possibilities.

A naming heuristic

- 3 step process
  - Propose a name
  - Three reasons why the name is not adequate
  - Propose another name that eliminates these problems
  - Repeat if necessary
- Try it together?

The Design….Finally!!

- Given our process for identifying a problem within a problem space, what do you think the first step in the design process is?

The fun part, your design!

- Given our process for identifying a problem within a problem space, what do you think the first step in the design process is?
- Generating ideas!

- You can use the same techniques we have talked about ad nauseam, brainstorming, context-free questions, etc.
- Note – while problem spaces are often very large, it is possible for them to not exist at all. For this reason (and other similar ones) feedback may occur from the exploration of the problem space to the requirements.

Ideas

- There is such a thing as a bad idea.
  - An idea is only good in that it solves the problem you have to solve.
  - Do not be afraid to move on from ideas.
  - Incredibly elegant ideas can be bad in a certain context.
    - Paraphrasing Berkun - E=mc2 is a great idea if you are trying to expand your understanding of the universe, but not all that helpful if you are looking for water in the Sahara.
  - Bad ideas can lead to good ideas.

- Think outside, inside, and otherwise anywhere near the box.
  - Berkun – do whatever you need to do to solve your problem.
  - “Hell, there are no rules here. We’re trying to accomplish something.” – Edison; quote from Berkun.

Quantity!
It is OK to have bad ideas, just be willing to give up on them.

“The physicist’s greatest tool is his wastebasket”
– Einstein

“There’s a better way to do it—find it.”
– Thomas Edison

“Fail. Fail again. Fail better.”
– Samuel Beckett

“If you want to succeed, double your failure rate.”
– Tom Watson, IBM

“I write 99 pages of shit for every one page of masterpiece.”
– Ernest Hemingway

Quotes Example from: “Exploring requirements – quali
Weinberg and Gause

Improvisation – one more ideation technique

Four rules:

1. Yes, and…
   - Whenever an idea is offered, the only acceptable response is yes, and <insert idea here>.

2. No half-assing
   - You are not allowed to say…"sorry...this isn't a great idea".

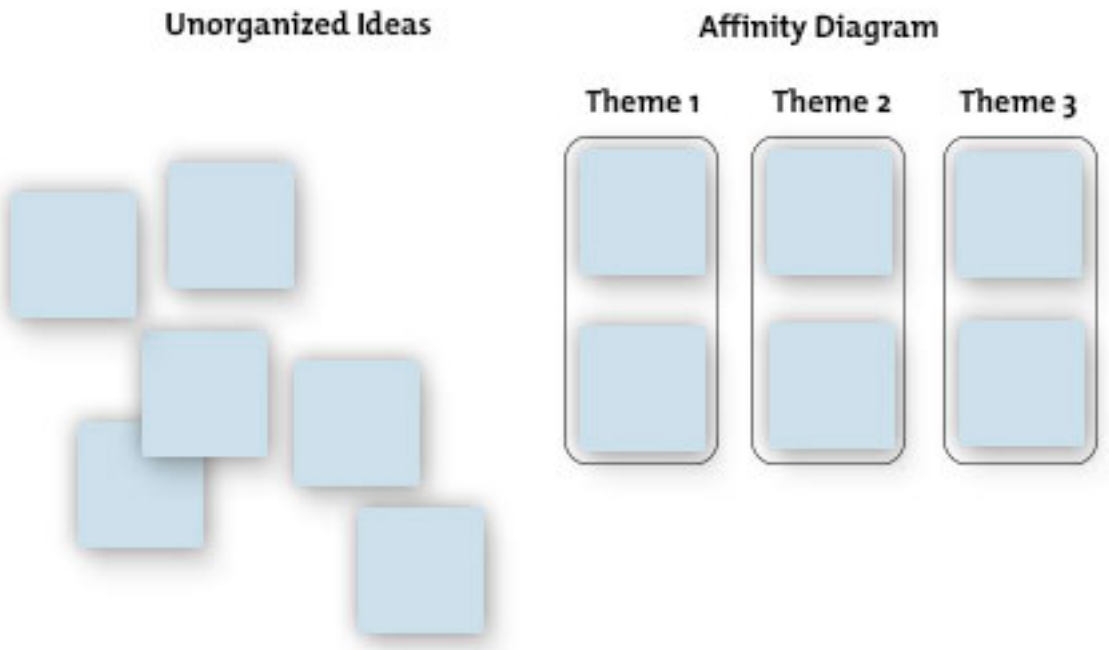
3. No blocking
   - You can't judge another person's idea negatively.
   - This stops the conversation.

4. Make your colleagues look as good as possible, we will offer kudos.

Quotes Example from: "Exploring requirements – quali
Weinberg and Gause

Affinity Diagram – reducing ideas

- Term invented by anthropologist Kawakita Jiro in the 1960s.
  - Studied remote Nepalese villages in the Sikha Valley.
  - Designed information analysis system to help villagers identify their most important problems.
- Group ideas based on some criteria, such as their applicability to a specific requirement.



https://www.mindtools.com/media/Diagrams/Affinity-diagram-final.jpg

Selecting a team

- No heuristics…
- Just common sense.

Topics we are out of time for

- Converting your RFA into an introduction.
  - Technical writing we will cover later in the semester
  - Investigating the market, likely to cover
- Feature Requirements
- Constraints

RFA – request for approval

- Apply what we have talked about to your own project.
- Step 1 – Post a problem space on the Web board!
  - Discuss the problem space with fellow students and the TAs to narrow it into a problem statement.
  - Once a problem statement has been identified, come up with requirements.
  - Once requirements have been specified, identify and discuss at least 3 solutions.
  - When that has occurred, I will ask you to post an RFA.
  - I will check once a day, probably in early evening.

- RFA – On the web board. Goal is to a rough outline of the project you will work on for the next 6 weeks or so.
  - Should include a project name, problem statement, high-level requirements, solution, and team members (2 to 3 people per project).