STANFORD HCI GROUP / CS247

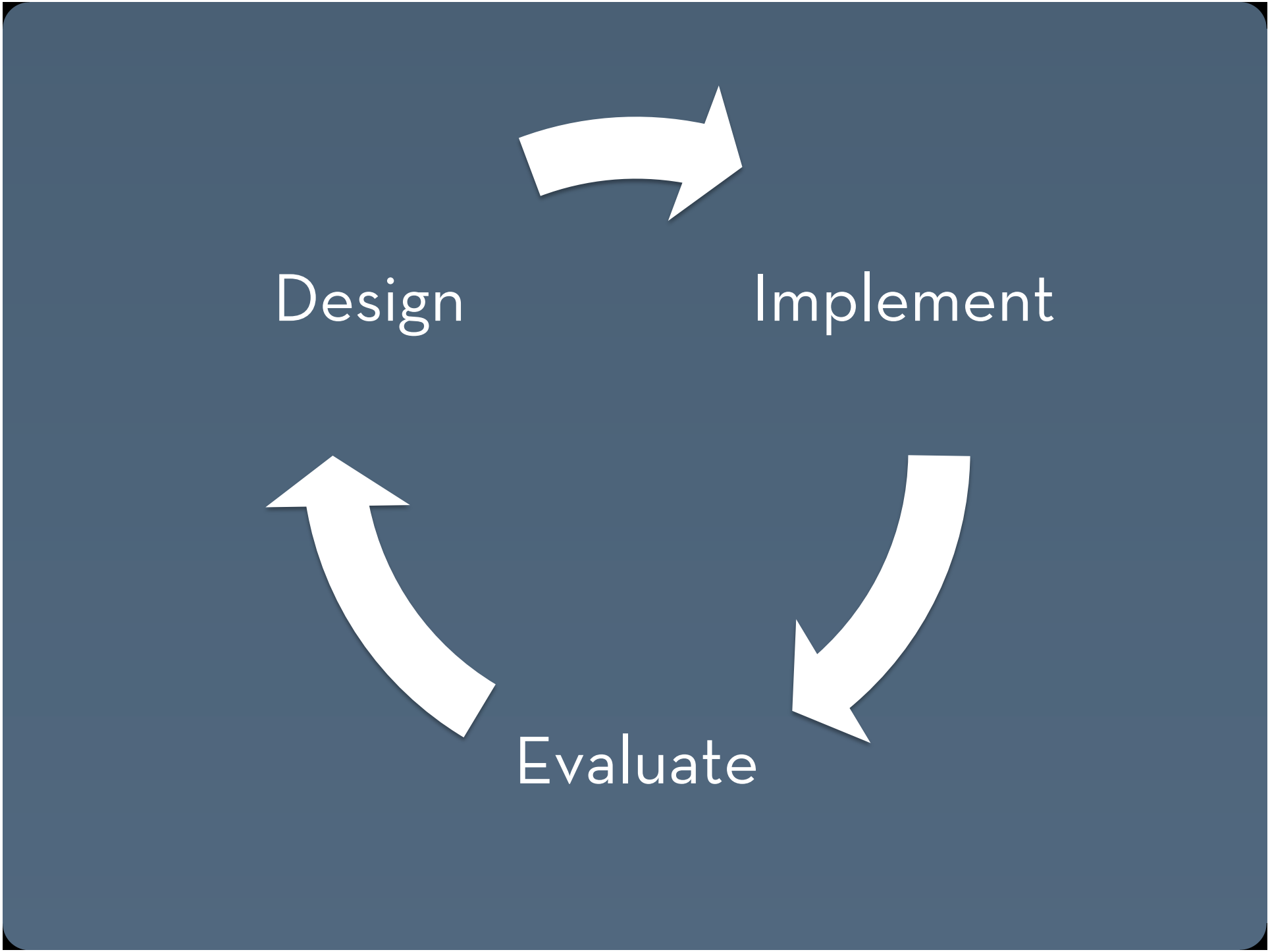
Human-Computer Interaction

Design Studio

12 February 2012

http://cs247.stanford.edu

OK, I built it.
Now what?


Design
Implement
Evaluate

What are bad reasons to evaluate a design?

Blind adherence to process: without a theory, you test the wrong aspect of the design.

Championing an idea: it's always possible to find *some* domain where your idea does best

Myopia: focusing on usability when you should be testing usefulness, or visa versa

What are good reasons to evaluate a design?

Usability: identify problems with a design and make local fixes.

Convince skeptics: understand which of multiple approaches will best serve users' goals

Identify floor and ceiling: demonstrate what your design does well, and what it doesn't

Approaches to Evaluation

Formative: Executed during the design process; offers course correction.

Summative: Executed at the end of a project; shows evidence that your design works.

Formative evaluations are for you.
Summative evaluations are for everyone else.

Formative Evaluation

Goal: Improve your design

The secret: give incomplete versions of your design to as many people as you can, as early as you can, as often as you can

Identify the minimum viable prototype to test your ideas at each stage

Rebutting the objection that you're already framing

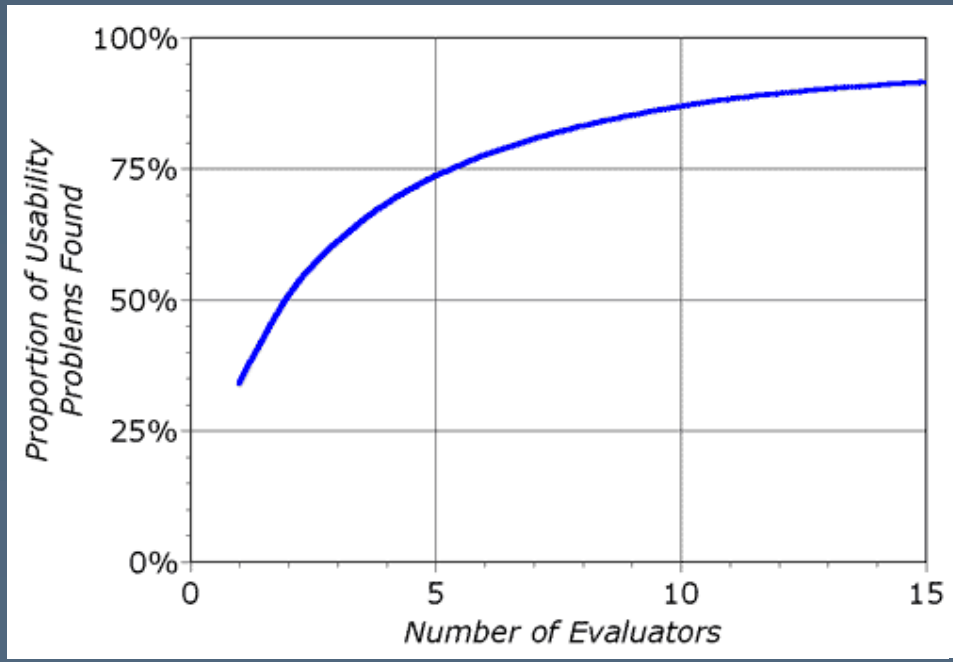
It is **not** so much work.

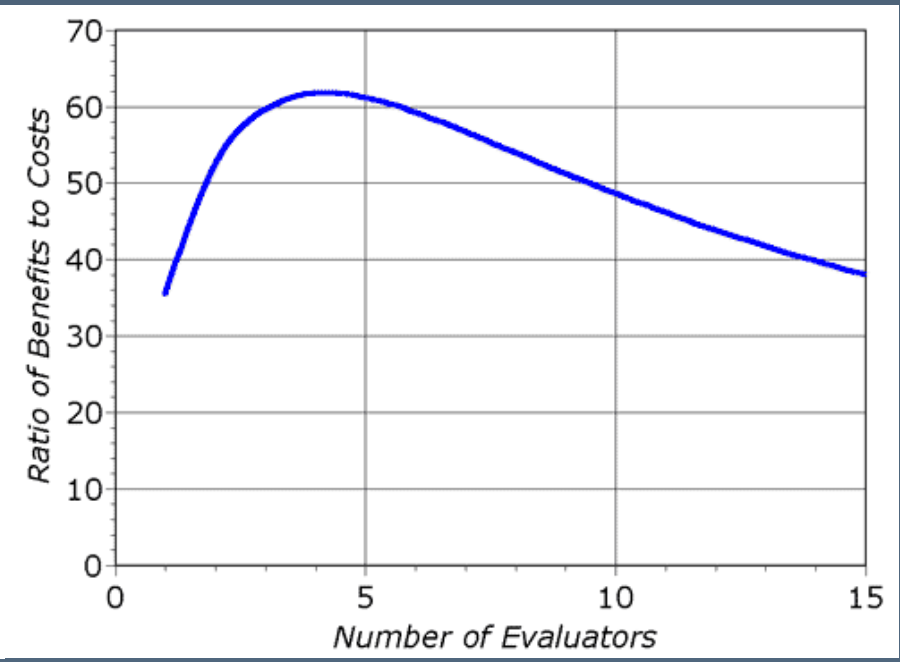
It is much, much less work than hacking on that next feature. You know, that one that will totally make your system way cooler.

I have never, ever seen an N+1 feature become decisive in a design's success or failure.

How little work is it, exactly?

How many evaluators do you need to catch usability flaws?





[Nielsen & Landauer 1993]

Four evaluators cost $6400 and find problems worth $400k

Usability Testing

Provide tasks (e.g., easy, medium, hard)

Talk-aloud protocol (users verbal reports)

Log usage

Pre/post study surveys

Never interrupt unless a user is truly stuck.

Wizard of Oz Testing

Prototype an interactive system by using humans to simulate machine behavior.

Focus your efforts entirely on the interaction, and fake the glue code and algorithms.

Act exactly as a computer would, including any recognition errors.

Wizard of Oz Demo

Volunteer?

Wizard of Oz Demo

I have designed a gestural system that advances slides in this lecture. A pointing gesture triggers a click on the spacebar.

Unfortunately, if I gesture in a way that looks vaguely like a point to the Kinect, it triggers the click as well.

A point back over my shoulder rewinds the slides.

The Wizard is not allowed to talk.

How to make a WOz Prototype

Map out scenarios and application flow: what should happen in response to user behavior?

Build interface skeletons (minimal autonomy)

Develop hooks for wizard input. For example, the wizard might react by change the interface by clicking on a link on another screen.

Rehearse wizard role with teammates.

WOz Testing on Thursday

Come with your group, prepared to setup and conduct a series of WOz tests.

Members of your group will rotate out to serve as test participants for another group.

Rehearse ahead of time!

Start building your software! Next week you will show your “bare bones” functional prototype.

Summative Evaluation

How would you evaluate...

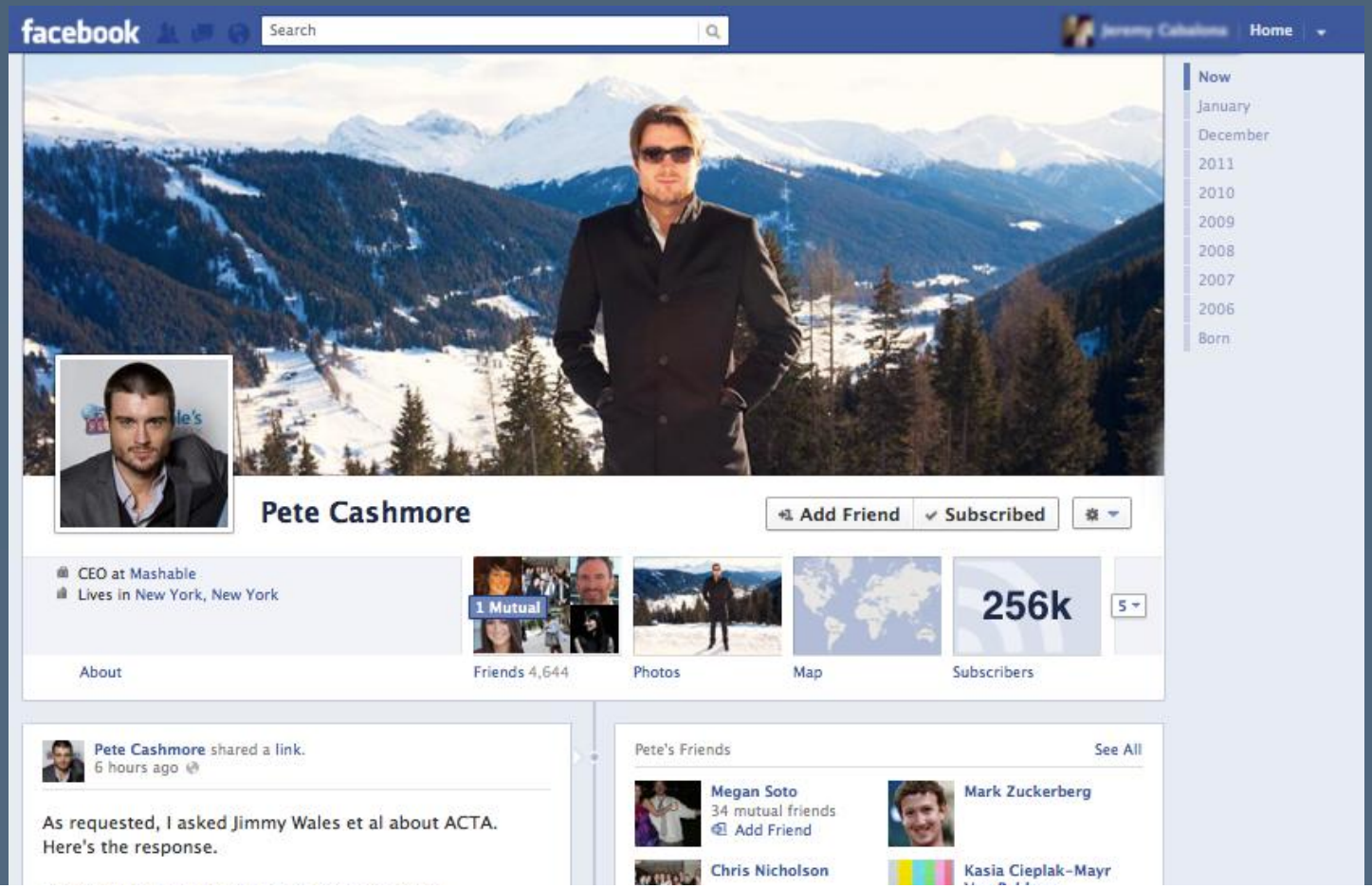
Apple's Knowledge Navigator?

How would you evaluate...

Snibbe's Visceral Cinema?

How would you evaluate...

The Facebook Timeline?



Framing an evaluation

The difficulty: defining and isolating the construct that you are trying to maximize

Tempting to aim for something easy: time, task completion, number of clicks

But, testing the easily quantifiable often **misses the point**.

Framing an evaluation

Reflect on your implicit thesis about why your design is great.

> The Apple Navigator is great because...
> Visceral Cinema is great because...
> The Facebook Timeline is great because...

This thesis can directly imply the claim that you need to test in your summative evaluation. (It may or may not be comparative in nature.)

Example theses

Enable previously difficult/impossible tasks

Improve task performance or outcome

Modify/influence behavior

Improve ease-of-use, user satisfaction

User experience

Practicing with P4

One sentence: what is the implicit thesis that explains why your design is great?

One sentence: to understand whether that thesis is true, what hypothesis do you need to test?

Methodology Matters

Method Toolkit: Mix and Match

Proof-of-concept system

Inspection (Walkthrough) Methods

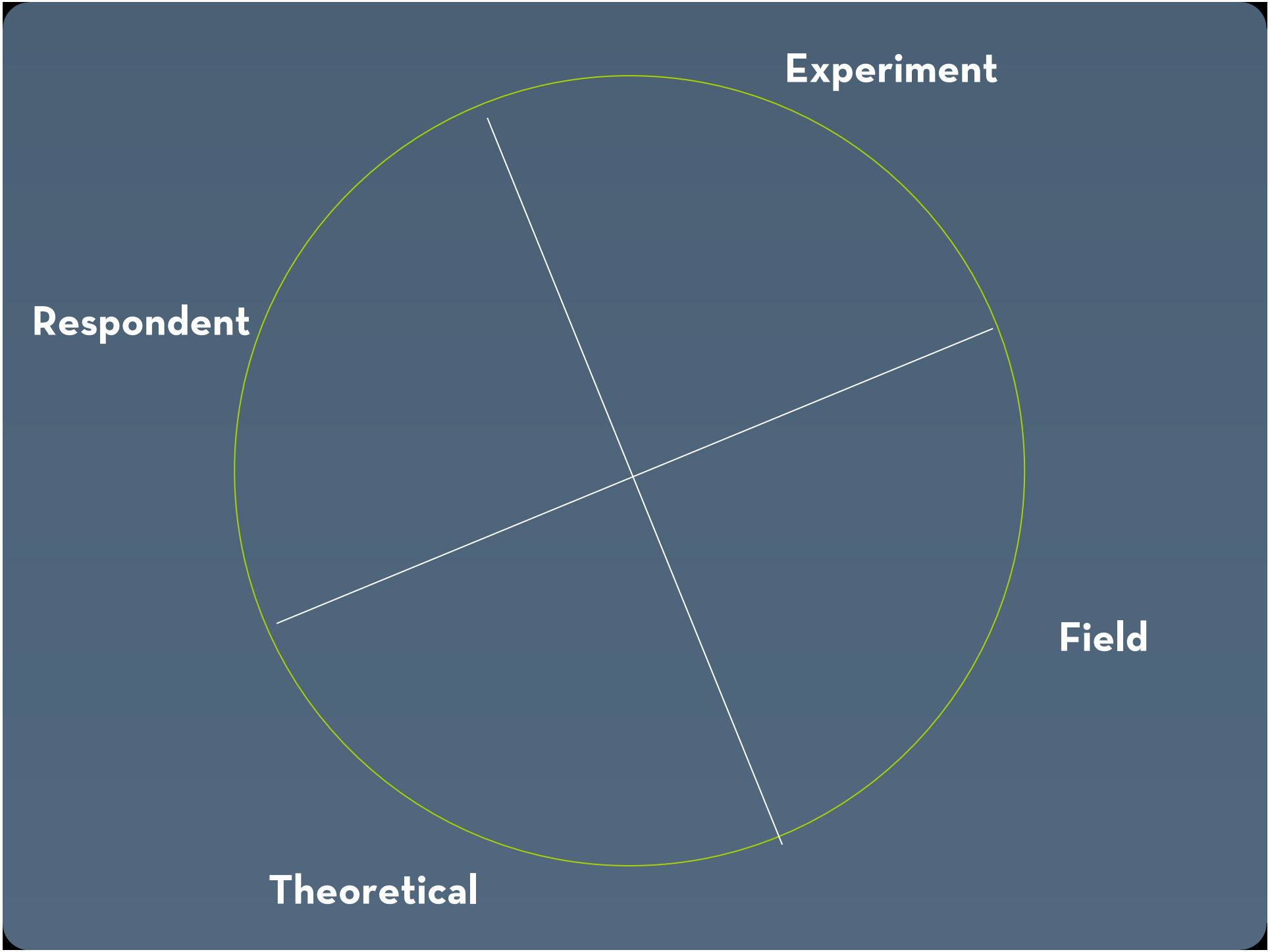
Observation, User Studies

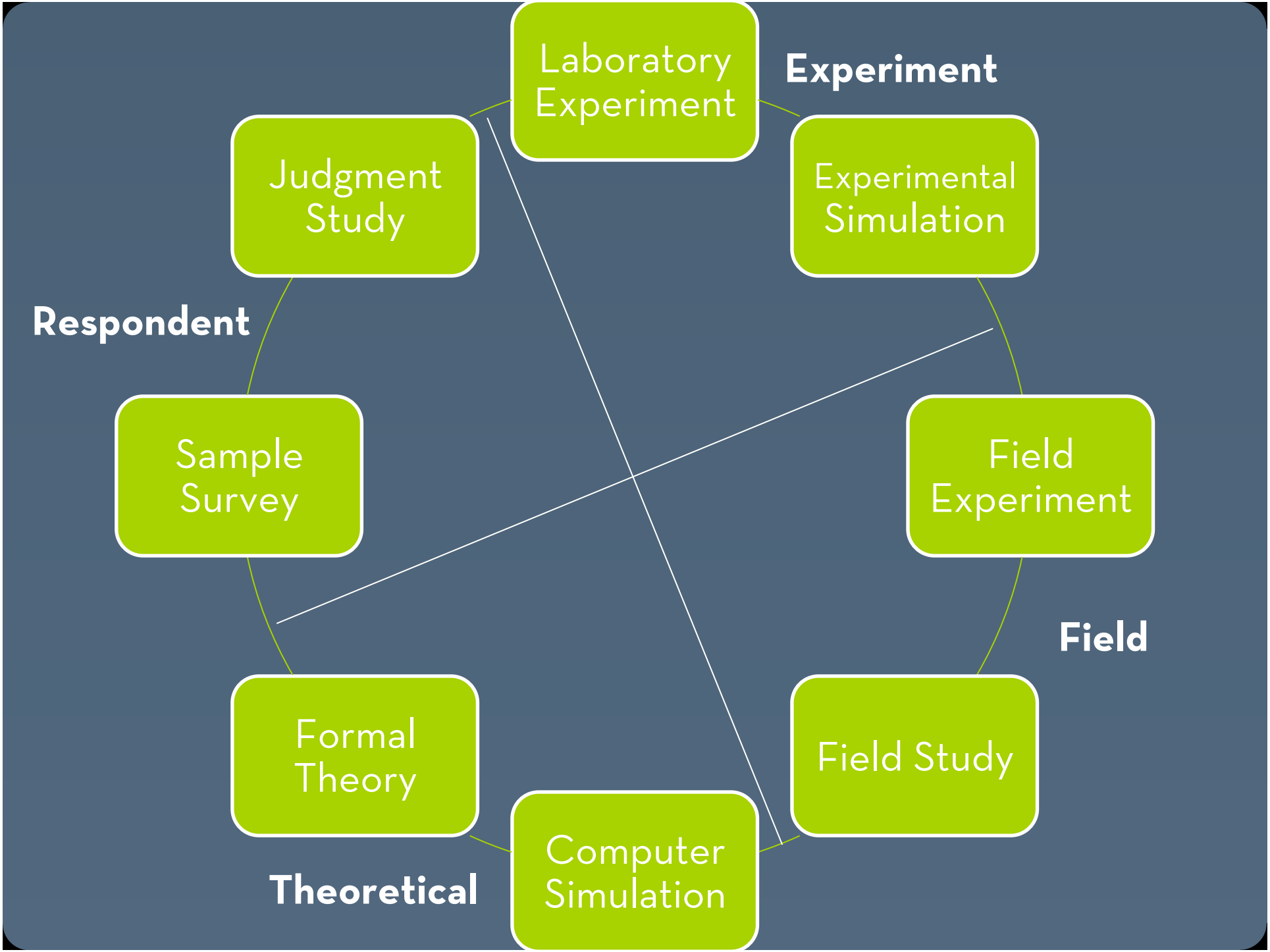
Interviews and Surveys

Usage Logging

Controlled Experimentation

Fieldwork, Ethnography


Experiment
Respondent
Field
Theoretical


Laboratory Experiment
Experiment
Judgment Study
Experimental Simulation
Respondent
Sample Survey
Field Experiment
Field
Formal Theory
Field Study
Computer Simulation
Theoretical

Controlled Experiments

Manipulate *independent variables (IV)*, measure *dependent variables (DV)*.

Within or *between-subjects* design

- Change IVs within or across subjects

Randomization, replication, blocking

Learning effects

Subject population, number of participants

Experimental Desiderata

Choice of measure and statistical tests

t-test, ANOVA, Chi-squared χ^2,
non-parametric

p-value: probability that results due to chance

Type I Error: accept spurious claim
Type II Error: mistakenly reject claim

Statistical vs. practical significance

N=1000, $p < 0.001$, avg dt = 0.012 sec.

Internal Validity

Is a causal relation between two variables properly demonstrated?

In other words: did you prove it?

Threats to internal validity: confounds, improper randomization, experimenter bias

External Validity

Do the results generalize outside of the study?

In other words: does your proof apply to real situations?

Threats to external validity: participant population, task choice, training, payment

Proof by Demonstration

Prove feasibility by building prototype system

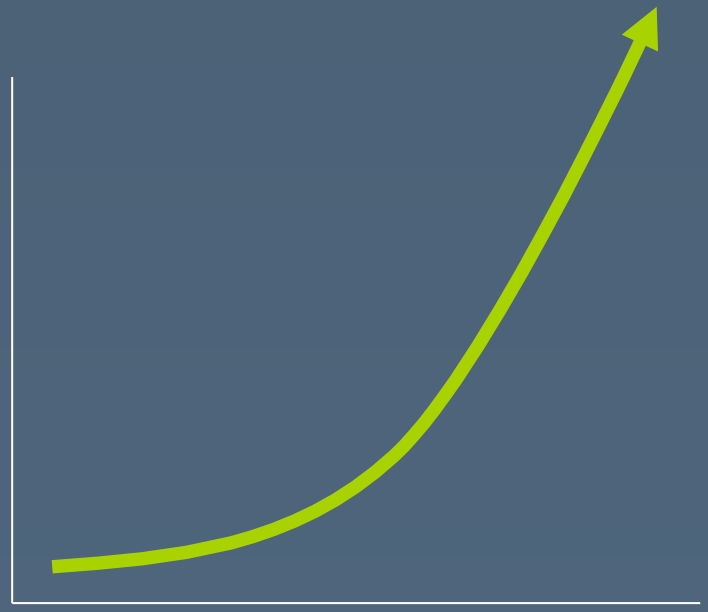
Demonstrate that the system enables task

Important to demonstrate the floor and the ceiling of your system's abilities. Be up-front about both.

Proof by Purchase?

"The design is good because it succeeds in the market."

Consider: "Windows 95 is designed well because it's the market leader."



Limitations: neither a necessary nor sufficient condition

Are there benefits to this approach?

In sum:

Evaluate continuously during the design process to spur iteration. If it's hard to do, make it easier.

Evaluate at the end to test the core thesis that inspired your design.

Choose the evaluation method that matches your claim.