

Your product may be worthless

and other uncomfortable
truths about software
development

Jeff Patton

AgileProductDesign.com

jpatton@acm.org



1. **Software is risky** business
2. Software **process** is more distraction than solution
3. **Practices** that get you into the good product business




Part 1: Risky Business

Process makes me anxious.

Isn't there something we're overlooking?





Typically about 50%
to 80% of all software we
ship fails to accomplish it's
objectives.

People like Marty say this stuff is hard

(Marty Cagan, author of Inspired, How to Create Products Customers Love)

Return on investment for
software is far from certain



Source: Stevens, G.A. and Burley, J., “3,000 Raw Ideas = 1 Commercial Success!”,
(May/June 1997) Research Technology Management, Vol. 40, #3, pp. 16-27.

For every 4 projects that enter development, only 1 makes it to the market

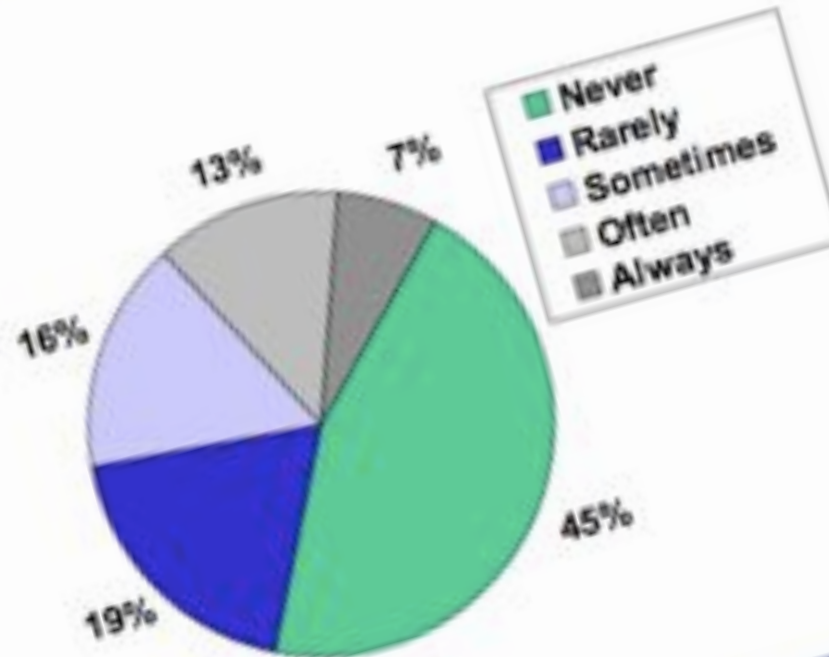
At launch, at least 1 of 3 products fail despite research and planning

An estimated 46% of all resources allocated to product development and commercialization by U.S. firms is spent on products that are cancelled or fail to yield an adequate financial return.

Source : *Winning at New Products*, p.9

Features & Function Usage

THE
STANDISH
GROUP



64% of features and functions are rarely or never used?

It's only after delivery that we understand value

opportunity:
integrated music
management and
portable music
player



“There were plenty of weak spots that led to Microsoft's disastrous December quarter, but one that didn't get much attention Thursday was how badly the Zune did.”

--Ina Fried, CNet News,
January 2009

high
demand, high
value solution:



low
demand, low
value solution:

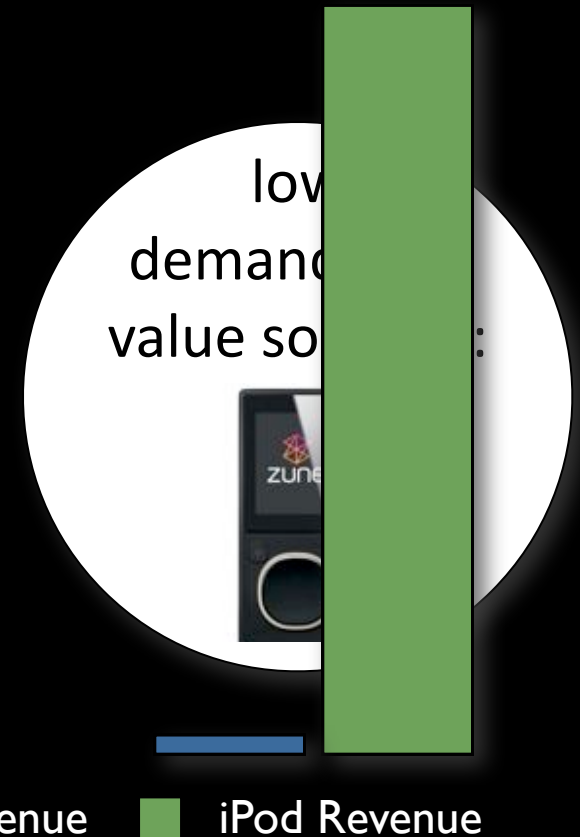


What's the business value for the same feature Apple and Microsoft's backlog?



"Zune platform revenue decreased \$100 million, or 54 percent, reflecting a decrease in device sales,' Microsoft said. That's quite a drop."

Apple, by contrast, saw its iPod unit sales up 3 percent, while revenue dropped by 16 percent. It still racked up \$3.3 billion in revenue, as compared with less than \$100 million for the Zune."



■ Zune Revenue ■ iPod Revenue




Prioritize by business value?


Is that really your best advice?



Part 2: It's a Racket

A man in a dark suit, light pink shirt, and dark red tie is holding a yellow banana to his ear as if it were a telephone. He has a serious, slightly angry expression.

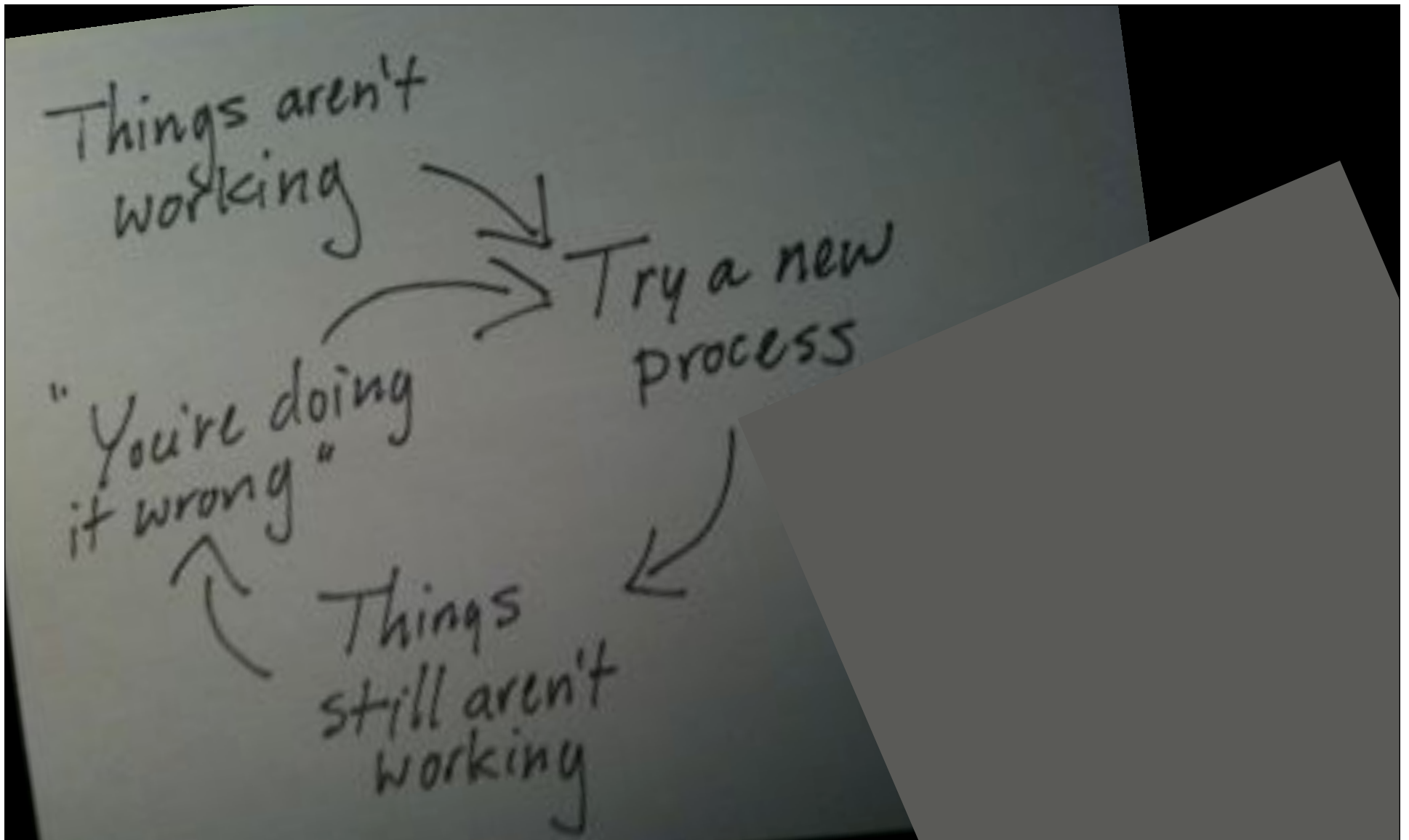
Hello... our customers
are complaining because they
hate our software!

A man in a dark suit, white shirt, and red tie is smiling broadly, showing his teeth. He has a confident and optimistic expression.

We believe if we could
ship more of it faster, that would
solve our problems!

I've got a process
that'll fix everything!

It starts with a process that'll solve
your problems



The popular process improvement cycle

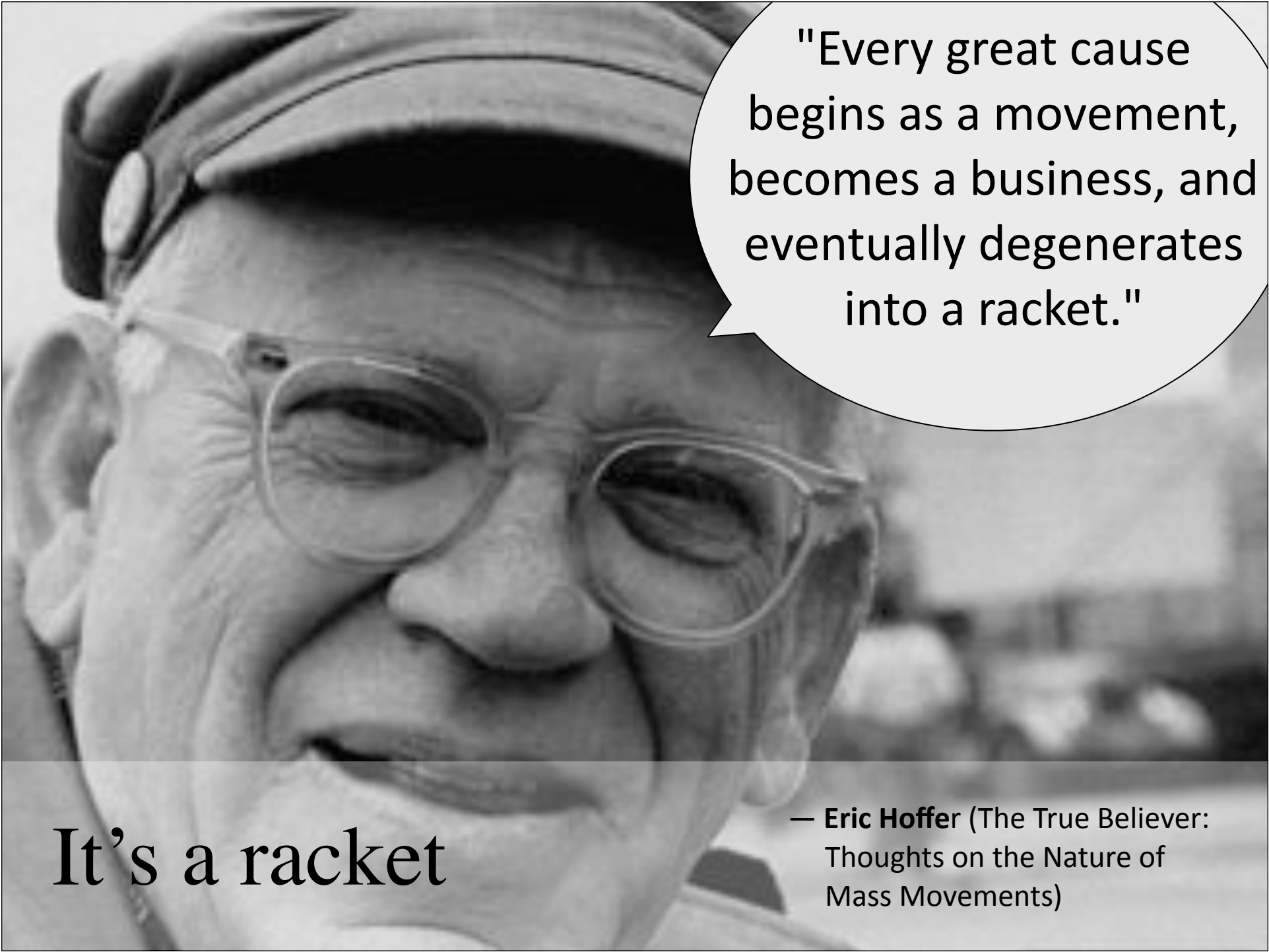
Try the Nokia (Scrum-but) test to see if you're doing Scrum wrong...

1. Fixed iteration length of 4 weeks or less
2. Software is tested and deployed each sprint
3. Good user stories tied to specifications as needed
4. A product owner with clear product backlog estimated by the team before the sprint planning meeting
5. Product owner can measure ROI based on real revenue, cost per story point, or other metrics
6. Estimation error < 10%
7. Burndown only burns down when story is done
8. No one is disrupting the team, only Scrum roles

Source: jeffsutherland.com/nokiatest.pdf

Note that this is just portion of the test. Download the pdf to see the full text

Does your process
make your but look
big?



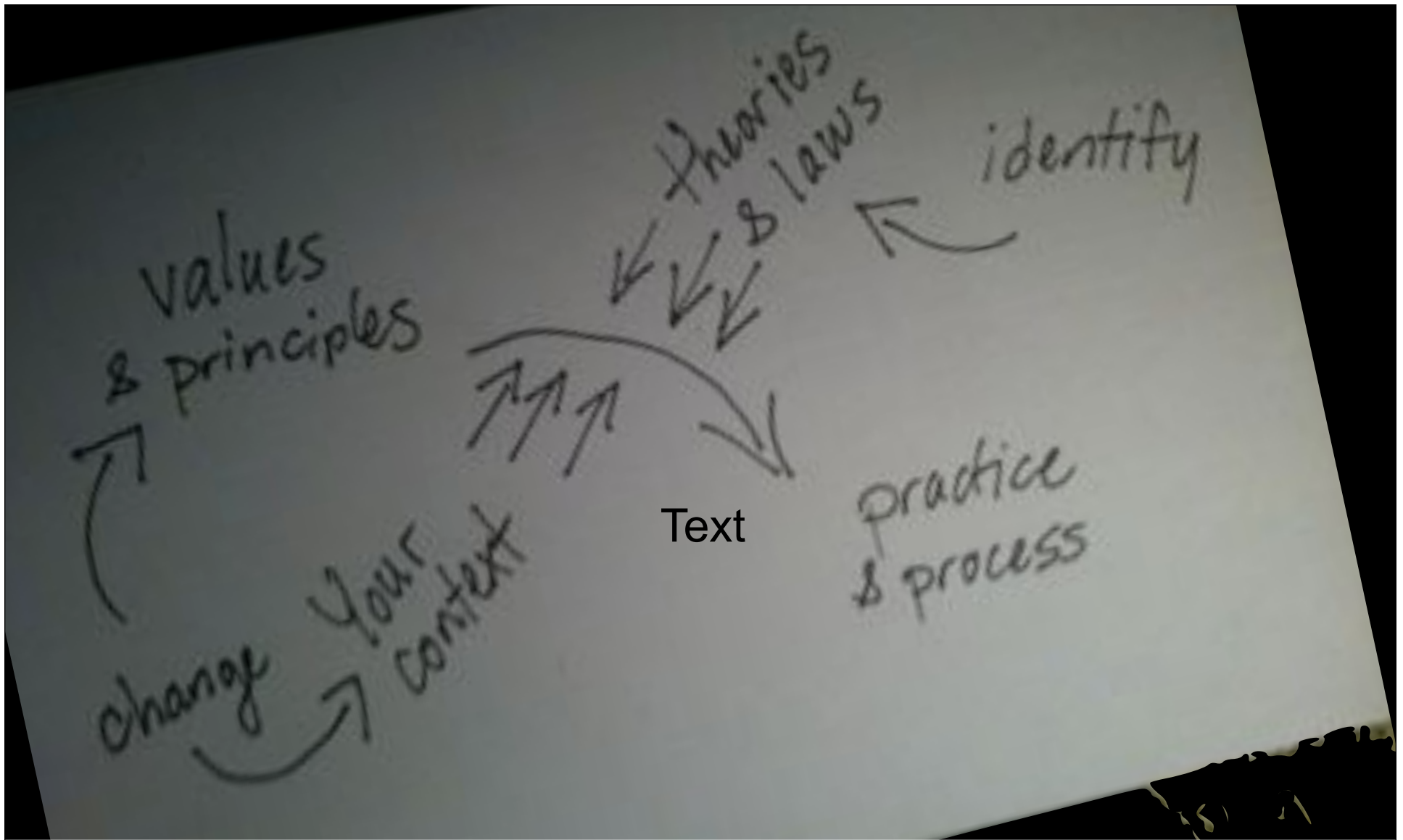
"Every great cause begins as a movement, becomes a business, and eventually degenerates into a racket."

It's a racket

— **Eric Hoffer** (The True Believer: Thoughts on the Nature of Mass Movements)



Processes are like haircuts.
Copying someone else's is usually a bad idea.



The only process that can work is yours

Derived from Alistair Cockburn's Crystal Model: <http://alistair.cockburn.us/Crystal+3-Step+Model>


1. Look to agile values and principles, can your organization support them?
2. Add your organization's values and principles
3. Start with Scrum, XP, or other agile process frameworks
4. Keep an eye out for other practices that can support your process
5. Take ownership of your process
6. Nurture your process, keep an eye on change, keep learning



Part 3: Getting your hands dirty



Juarez is a CEO who lost something with agile

A photograph of a man in a white polo shirt, looking to his right. A speech bubble is positioned over the right side of the image, containing text. The background is split into a light grey left half and a black right half.

We used to
take real pride in the
quality of what we built. Now
all I hear about is
acceptance criteria.

The agile manifesto doesn't include
Juarez's values

For many companies, their
software is part of their
brand, part of who they are.

Value building software you're
proud of.

The forgotten law:

Software is hypothesis

We don't know if it's valuable
until it ships

Stop guessing
Start learning

Start with users:
Stop talking about users,
start talking to users.

Rethink user on site

← a user proxy - someone who used to be a user

← users) on site

← frequent visits to a diverse group of users

← a large pool of available users to collaborate with

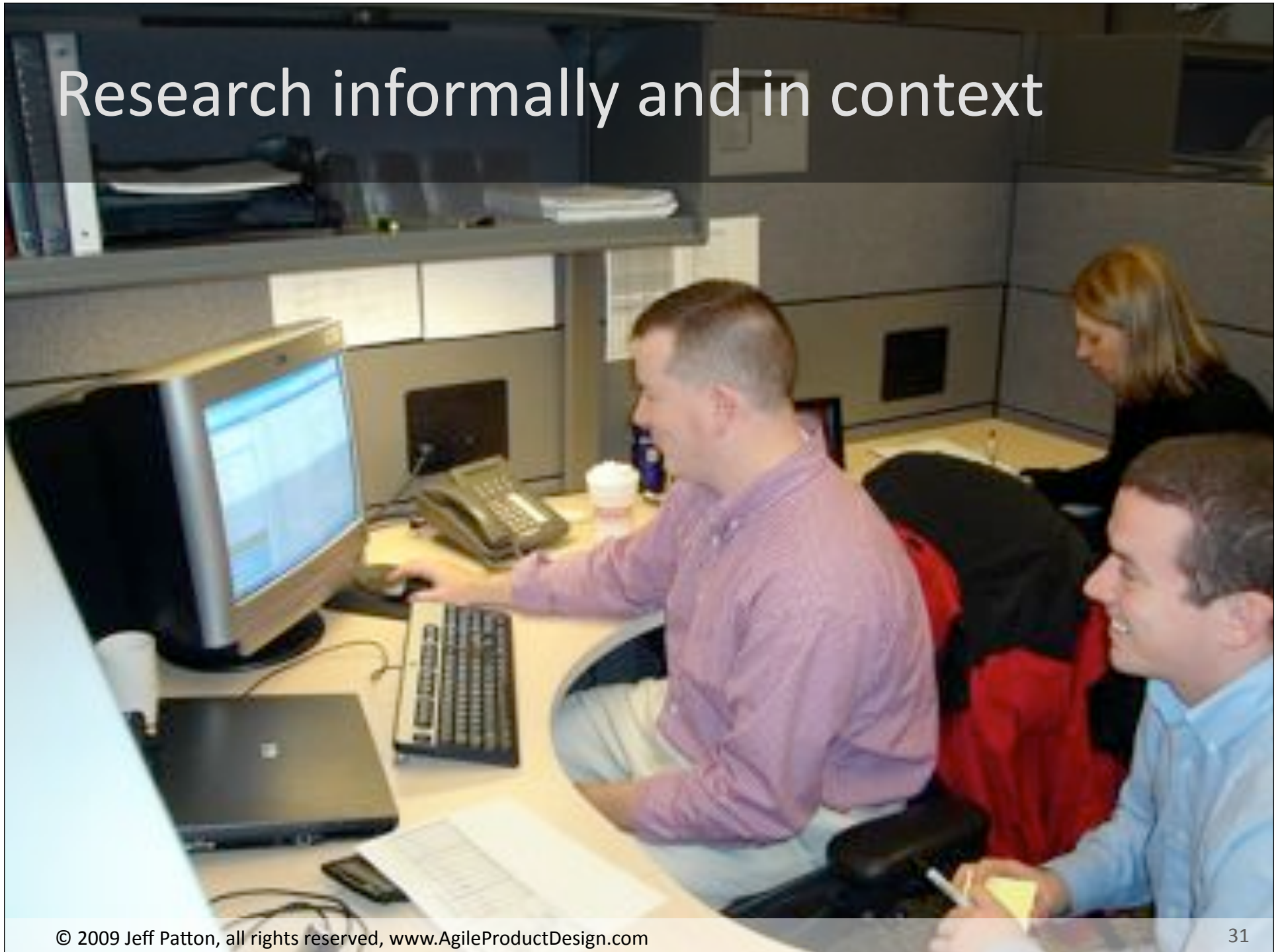
“The team working on NEC’s PC 8000 consisted of sales engineers from the Electronic Devices Division.”

“They acquired much of the know-how to develop the company’s first personal computer by putting together TK 80, a computer kit, and introducing it on the market two years in advance of the PC 8000;
and

by stationing themselves for about a year, even on weekends, at BIT-IN, an NEC service center in the middle of Akihabara, talking with hobbyists and learning the user’s viewpoint.”

-- Hirotaka Takeuchi and Ikujiro Nonaka, The New New Product Development Game

Research informally and in context



Bring users and their experience back into the team

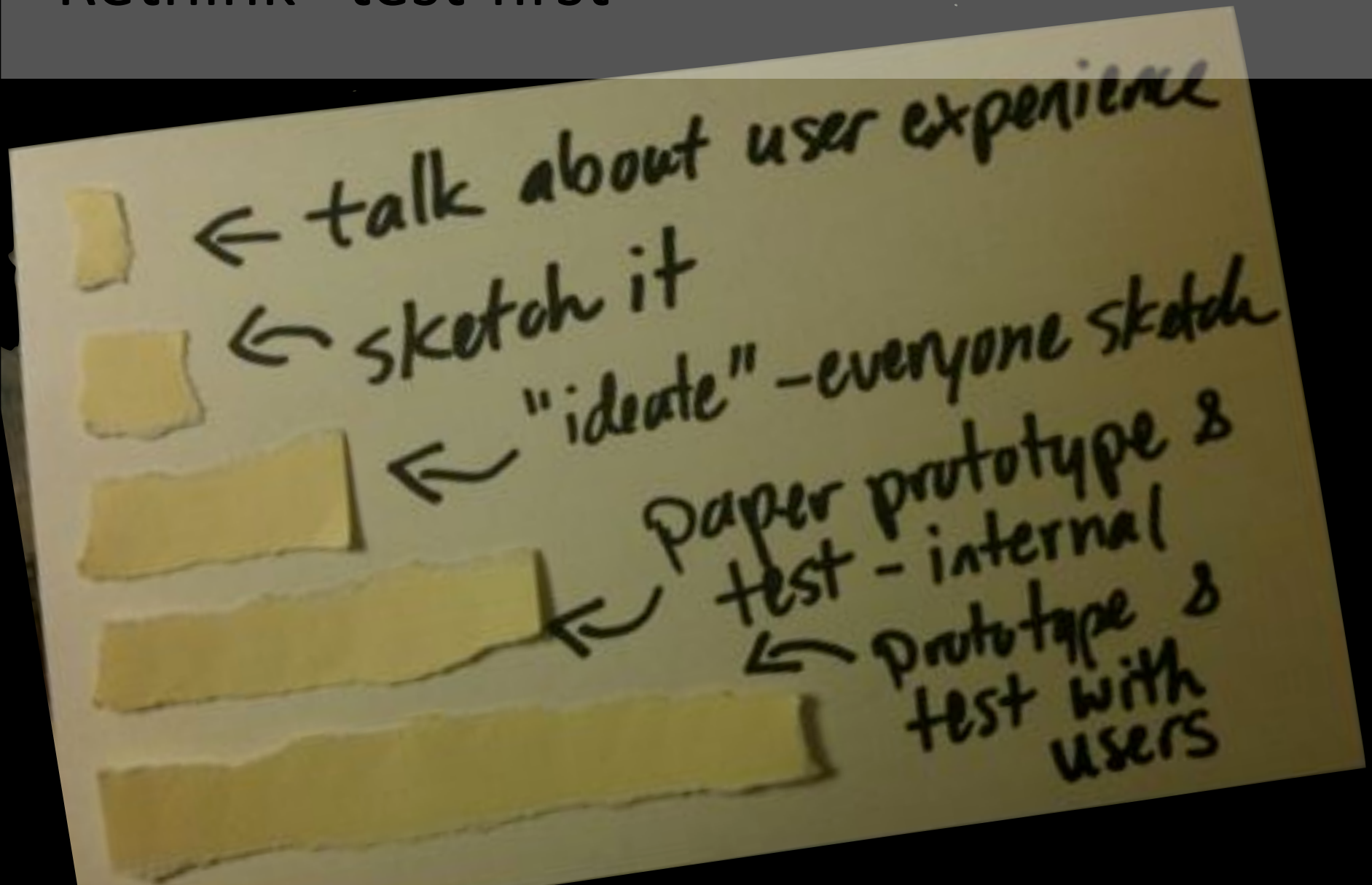


Collaboratively create lightweight and relevant pragmatic personas



Replace big design up front
with **big learning up front**

Rethink "test-first"



Sketch UI Collaboratively



Sketchboarding is collaborative way to come up with UI ideas



Practices like Design Studio and Sketchboard help us collaboratively design and ideate



Design Studio Approach

http://interaction08.ixda.org/Jeff_White%20and%20Jim%20Ungar.php

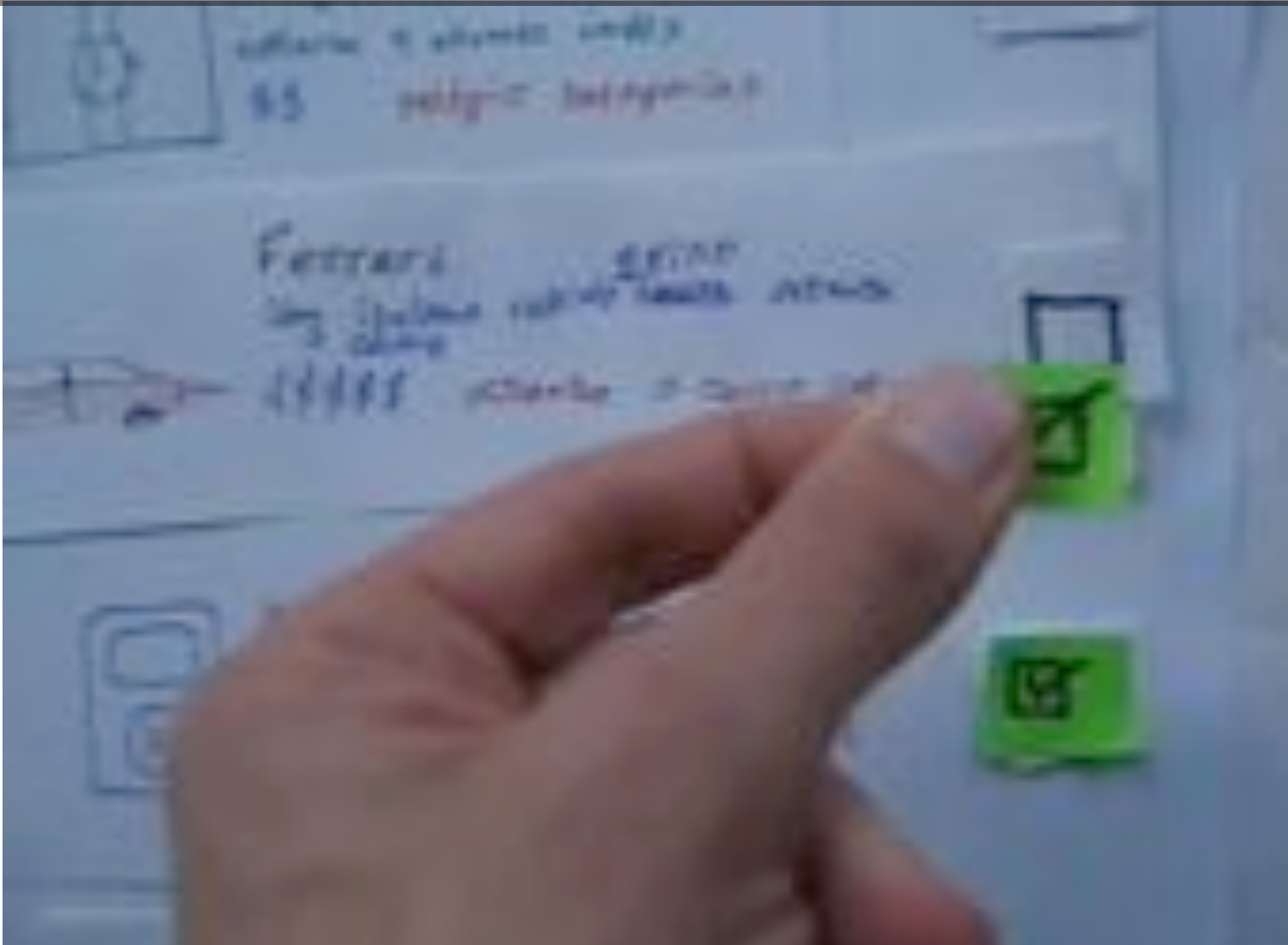
Sketchboarding

<http://www.adaptivepath.com/ideas/essays/archives/000863.php>

Communicate user experience



Joe recommends using the paper prototype to communicate the UI design



Build a **story map** to describe the user experience



Replace product show-and-tell
with genuine product
evaluation

Increase validation to reduce delivery risk

← Sprint review/product demo

← we evaluate assuming role of our user

← users evaluate by performing tasks - we observe

← users use it in context

Testing means really using the product to validate you can accomplish something



It's the product, stupid

Own your process

Choose practices that help you
get the product you want

Your product may be worthless

and other uncomfortable
truths about software
development

Jeff Patton

AgileProductDesign.com

jpatton@acm.org

