Deploying SW Defect Detection Tools

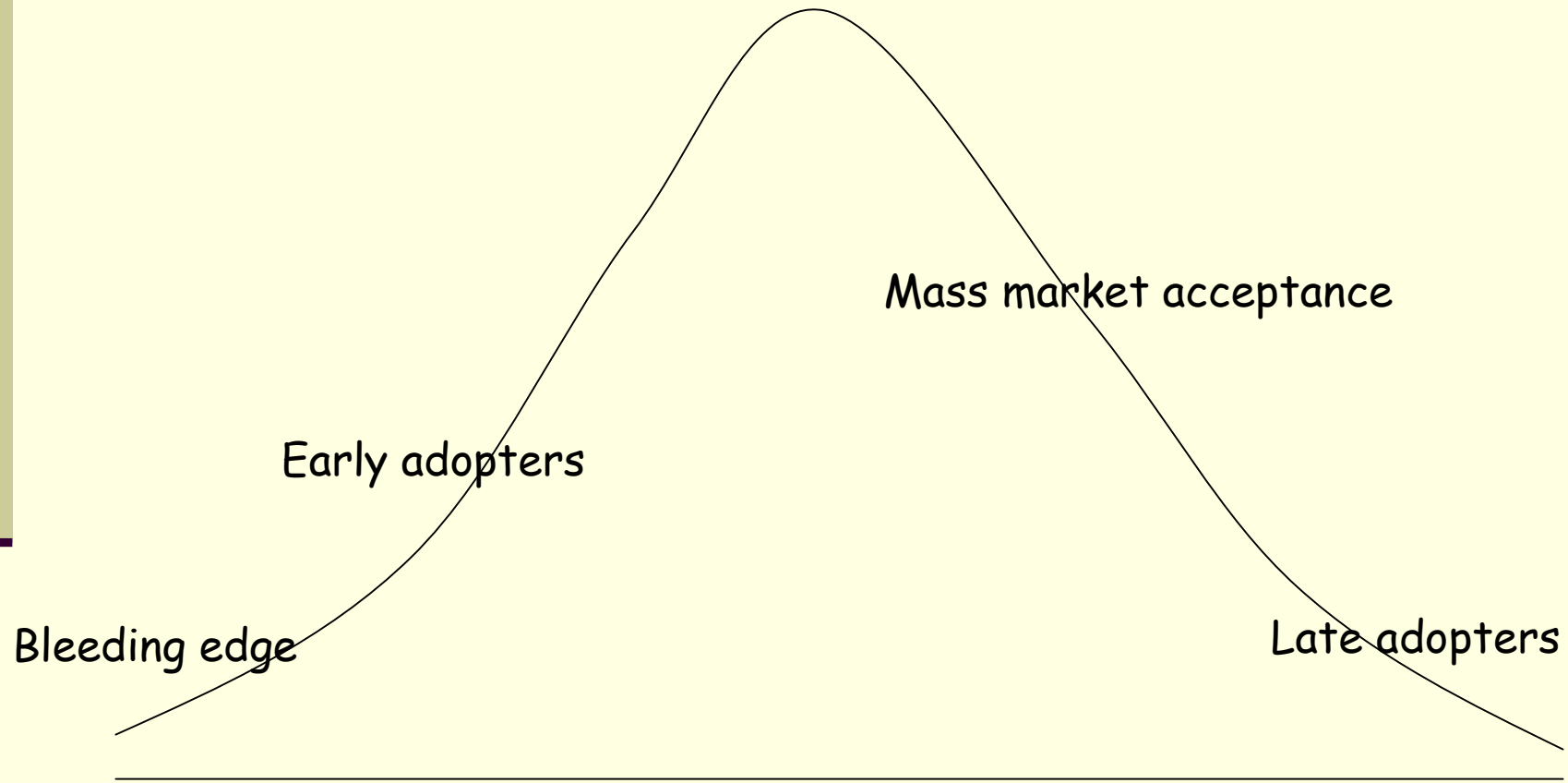
David R. Cok
Eastman Kodak R&D
June 2005

To encourage adoption you must

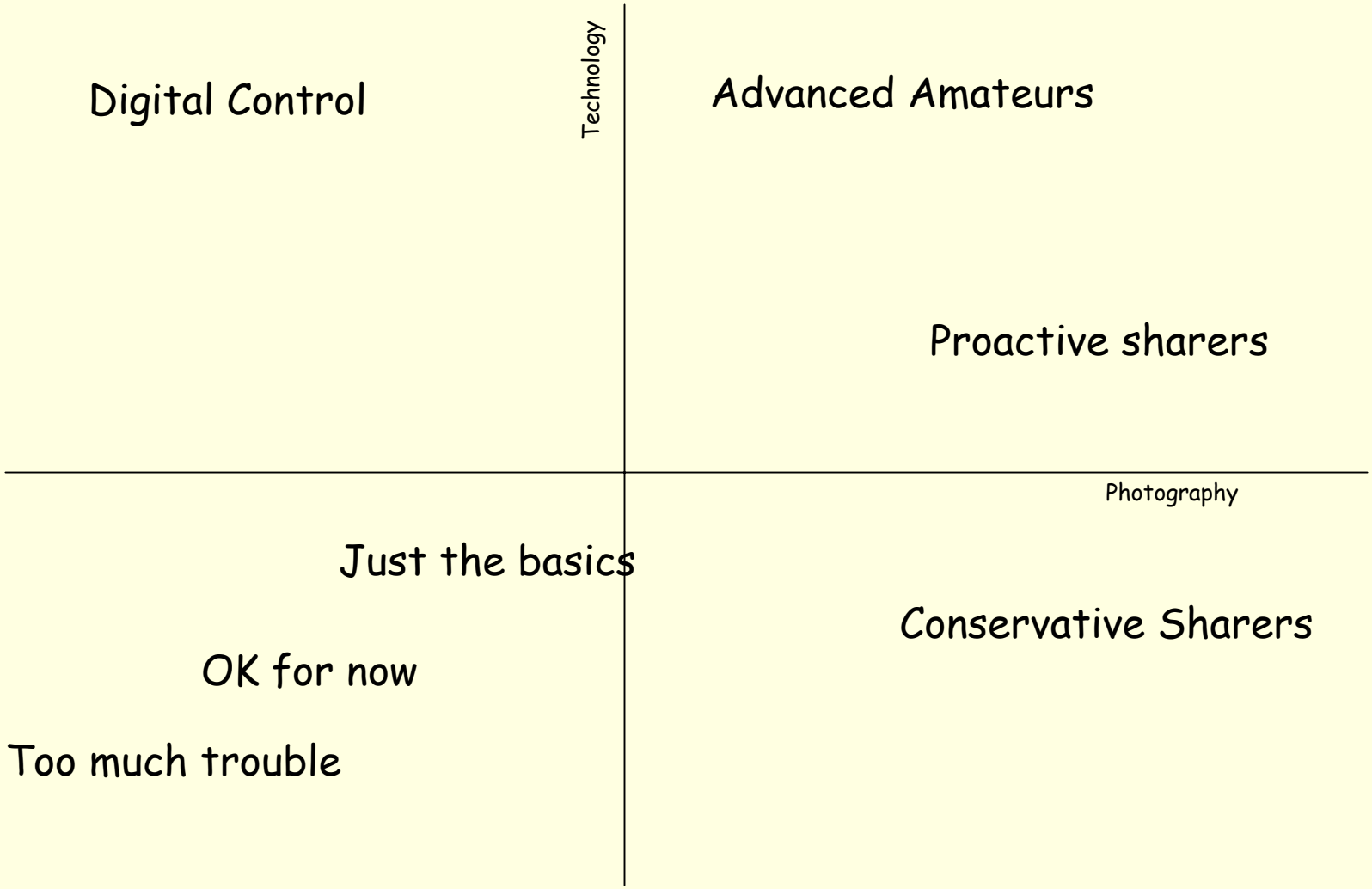
understand your customer
(and that might take some research)

Typical strategies

- Segment the target population
- Understand the differing user needs and motivations within the subpopulations
- Address those needs with appropriate features, services and price points


Mass market acceptance
Early adopters
Bleeding edge
Late adopters

Photography business…



In every category

... it is all about ***Ease Of Use*** ...

just at different thresholds

SW tool users

- Early adopters
  - follow technology, interested in new technology, tries new things (a little bit)
- Mass market
  - busy doing "real work"
  - new tools must
    - provide real value
    - do so "cost effectively"

Three dimensions

- Usefulness
- Usability
- Good long-term investment
- (+ sociological, organizational dimension)

Usefulness

- Reasonably accurate in its output

Truth \ Detector	Says it's not a bug	Says it's a bug
It really is a bug	False negative (FN)	True positive (TP)
It really is not a bug	True negative (TN)	False positive (FP)

TP/(TP+FN) = recall = sensitivity = Pr(detect the bug | there is a bug)

TP/(TP+FP) = precision = predictive value positive = Pr(real bug| detected bug)

TN/(TN+FP) = specificity = Pr(no detection | no bug)

FN/(FN+TN) = predictive value negative = Pr(no bug| no detection)

There is experimental work to be done in measuring detection performance for a tool (and in comparing across alternatives).

Usefulness

- The tool must address an important defect area
  - reasonably frequently occurring
  - high value (difficult to find, high impact if not found)

- Some experiments with actual programmers and actual bugs would be helpful to characterize "bug importance"

Experimental work here too.

Usefulness

- Performance
  - Time and space performance is consistent with the value obtained
  - Does it scale with project size

Usability

- Out of the box experience
  - Simple installation & configuration
    - self-contained?
    - relies on other tools that need installation first?
    - multi-platform?
  - "Obvious" to use
    - quick to provide first benefit?
    - self-explanatory to a novice?
  - Does it do enough? but not too much?

Usability

- Resource investment (time,learning,money)
  - small investment provides some gain
  - increasing investment provides increasing gain
  - Can I obtain benefit on a small part of a large project?
  - Do I have to convert my whole project to a new environment?
  - Can I use this within/alongside of an existing work process or IDE?

Usability

- Perspicuity
  - Is the output clear?
  - Do the results actually save work?
  - Do I have to redo work each time I reuse the tool? (e.g. can I check false positives just once)

Long-term investment

- Is the investment in learning a new tool (or even paying for it) sound?
  - continuing support, ongoing maintenance
  - confidence in the work of the providing organization
  - confidence in the longevity of the providing organization

…and all that is the first step

- There are also organizational, cultural, psychological, managerial reasons for adoption or non-adoption
- Individual tool vs. team tool vs. organizational tool

This is a separate, but worthwhile (and difficult) area of experimental computer science research.

Summary

- Developers will adopt tools... but ...
  - Pay attention to usefulness *to them*
  - Pay attention to usability *for them*
  - Pay attention to the investment cost *for them*
  - And be aware of the subjective and organizational factors affecting adoption
  - There is *research* to be done in understanding these factors for individual tools and in understanding tool adoption as a whole.