IBM User Research and Design

Scaling Usability in Terms of Requirements

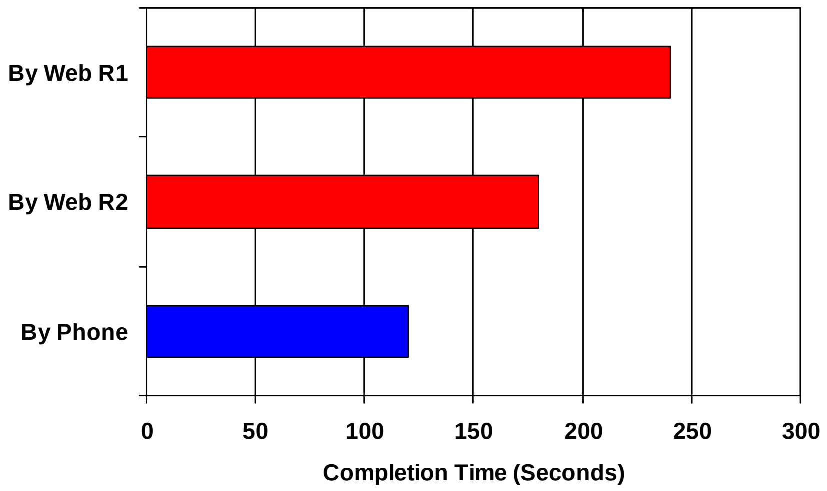
Assessing User Interfaces

IBM INTERACTIVE

Martin Fracker, Ph.D.

We measure usability in a variety of ways, but how do we know that usability is good enough?

Company XYZ is trying to redirect customers to perform transactions via the Web instead of calling into the Customer Service Call Center.



But the investment in a self-service website isn't paying off because Customers can make transactions faster by phone.

Usability engineering is about making software "easy to use" as measured by

- Fewer errors
- Faster task completion times
- Higher task completion rates
- Better user satisfaction ratings

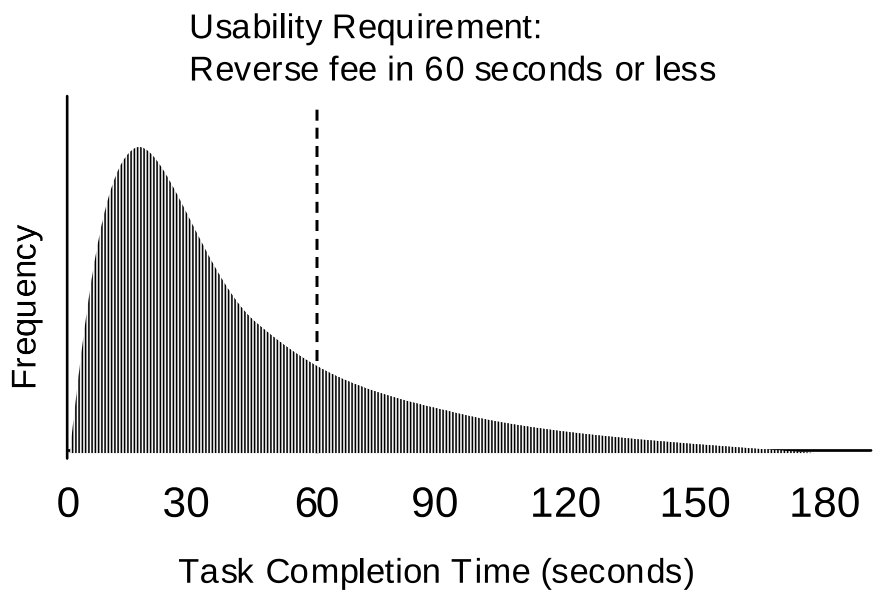
But how do we know that usability is good enough?

- Usability is always relative to something
  - An existing system
  - Alternative ways of doing the same things
  - Competitors' websites
- A "little bit more usable" may not be good enough

Suppose Company XYZ builds a customer self-service website to replace a Call Center

- Complete a typical phone transaction in 2 minutes
- Web Release 1
  - Same transaction takes 4 minutes
  - Customers therefore prefer to call instead
- Try again with Web Release 2: still not good enough

But the problem is often not so obvious. Sometimes, a usability metric can make a UI look like it's "good enough" when in fact it really isn't.



Distribution of Task Completion Times

Mean = 54 seconds

Positively skewed distribution

About one third of task completion times > 60 seconds

A bank expects CSR's to be able to issue a credit card fee reversal in less than 60 seconds.

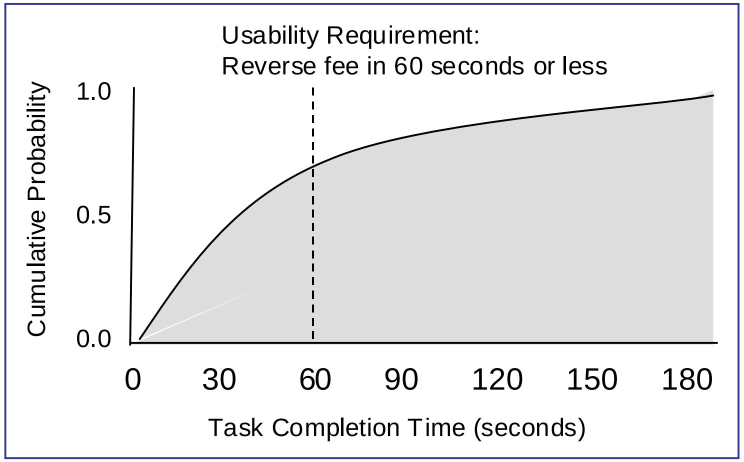
- Designs a new user interface for this transaction
- Conducts a usability test with six CSR's
  - Average transaction time: 59 seconds
  - Users complete fee reversal with no errors
- So the Bank
  - Builds the application
  - Deploys to thousands of customer service reps around the country

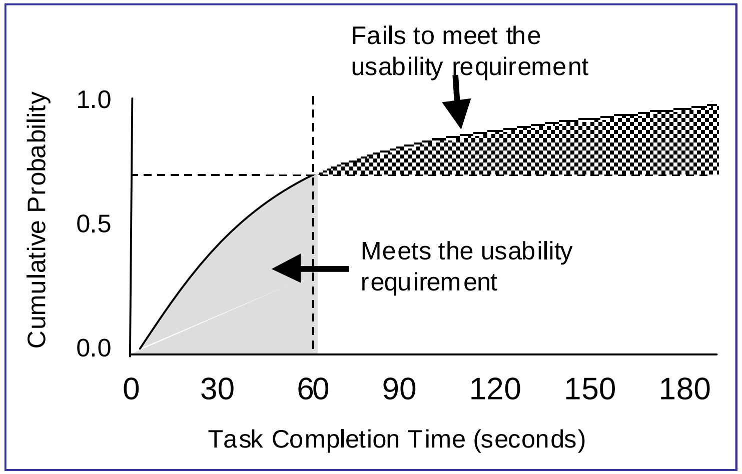
Six months later…

- Bank analyzes data from all call centers and thousands of CSR's
- Average transaction time = 54 seconds
- BUT
  - One third of all transactions > 60 seconds
  - The Bank was not expecting this
  - Wouldn't have built the app if they had known

Average completion time was the wrong metric!

The Bank SHOULD have measured the probability that the new application would meet the usability requirement of completing the transaction within the 60 seconds.





To estimate the probability that a user interface will satisfy a usability requirement

- We need to know the cumulative distribution of task completion times
- To determine the distribution:
  - Assume an underlying **Poisson process**
  - Implies distribution of completion times is exponential
  - Cumulative probability distribution is therefore given by

    $$P\ (T < t) = 1 - e^{-t/\theta}$$

    where

    T is total time to complete a task (a random variable)
    t is any arbitrary amount of time
    θ is the population mean of the distribution
- Evaluating the equation yields the probability that task completion time will be less than or equal to t

With mean = 54 seconds, P(T<60) = .67

Why a Poisson Process?

- Task completion times = waiting times for tasks to “arrive” at completion
- Completing one task is independent of completing other tasks
- The probability of two tasks completing at exactly the same time is zero

References: Taylor and Karlin, 1984, chapter 5; Hogg and Craig, 1978.

Using the cumulative probability distribution, we can determine what the mean completion time should be in order to meet the Usability Requirement

Probability that a user interface will satisfy the usability requirement to complete a task in 60 seconds or less for different values of the population mean (task completion time)

Mean	P (T ≤ 60s)
60	0.63
54	0.67
48	0.71
43	0.75
37	0.80
30	0.87
26	**0.90**

Suppose the bank decides that it will build the application only if t ≤ 60 seconds 90% of the time

- This implies that the average task completion time must be *26 seconds or faster!*
- If this is realistic, the Bank will need to keep improving the UI design until it meets this requirement
- If not realistic, the Bank may need to adjust its usability requirement by either
  - Accepting a lower probability of meeting the requirement
  - Adopting a less stringent requirement, e.g., 150 seconds

This analysis suggests a new way to scale usability

- Establish a usability criterion with two components
  - A target value for a metric (e.g., 60 seconds)
  - The percent of time this metric must be met (e.g., 90%)
- Estimate the average value of this metric with a usability test
- Apply the cumulative probability distribution to estimate the probability ***p*** of meeting the requirement
- Multiply the obtained probability by 100: **U = 100*p***
- **Compare U to the desired criterion (90%)**

Scaling Usability in Terms of Requirements

This usability scaling method can be applied to many of the things we typically measure in usability assessments besides task completion times: errors, completion rates, mouse clicks, and so on.

Most countable metrics are readily modeled by the Poisson distribution

$$P(X \leq n) = \sum_{x=0}^{n} \frac{\lambda^x e^{-\lambda}}{x!}$$

Where

X is the total number of events (a random variable)

x is any given number of events, $\leq$ n

n is the number of events observed or allowed

λ is the population mean

Examples of metrics that can be modeled this way:

- Errors
- Task completion rates
- Keystrokes or mouse clicks

Suppose Bank XYZ has three UI designs for a new application

- Usability Requirement
  - No more than 2 minor errors
  - 90% of the time
- The Bank conducts usability tests for all three
- Obtains the following mean number of errors:

Metric	Design A	Design B	Design C
Avg # Errors	1.0	1.17	1.5

- Applying the Poisson cumulative probability distribution, we obtain **U = 100P(X≤2)** for each design:

U	Design A	Design B	Design C
100*P(X≤2)	92	89	81

Only Design A meets the usability criterion, so the Bank adopts this design

Because our Usability Scaling method relies on estimating distributions from sample means, we must be concerned about the effect of small sample sizes on our estimates.

Signal Detection Table

	Truth	
Usability Scale Score	Meets Requirement	Fails Requirement
Meets Requirement	Hit	False Alarm
Fails Requirement	Miss	Correct Rejection

Hit	Correctly concluding that the usability requirement has been met.
False Alarm	Falsely concluding that the usability requirement has been met. The false alarm rate corresponds to the type 1 error rate in traditional hypothesis testing (Hogg and Craig, 1978).
Miss	Falsely concluding that the usability requirement has not been met.
Correct Rejection	Correctly concluding that the usability requirement has not been met.

So we did a series of Monte Carlo studies to evaluate the effect of small sample sizes on decisions made using the Usability Scale Score.

- Task completion times: 25,000 samples
  - From 25 distributions of task completion times
  - Distribution means from 23 to 122 seconds
  - 1000 samples of six observations from each
  - Six distributions met the usability requirement: 90% of task completion times < 90 seconds.
- User Errors: 25,000 samples
  - From 25 distributions of user error counts
  - Distribution means from 0.2 to 3.6 errors
  - 1000 samples of six observations from each
  - Six distributions met the usability requirement: 90% of tasks with < 3 errors

Non-parametric signal detection analysis for both Monte Carlo data sets

- **Sensitivity**: measures accuracy corrected for “random chance” (Pollack and Norman, 1964)
- **Bias**: measures tendency to favor either false alarms or misses over the other

The signal detection analysis yielded acceptable results for both sensitivity and bias, though the false alarm rates were higher than we would like.

25,000 Task Completion Time Samples

	True Probability	
U_t	≥ .90	< .90
≥ 90	Hits: .79	FAs: .16
< 90	Miss: .21	CRs: .84

Bias: β' = .94
Sensitivity: A' = .89

25,000 User Error Samples

	True Probability	
U_e	≥ .90	< .90
≥ 90	Hits: .78	FAs: .12
< 90	Miss: .22	CRs: .88

Bias: β' = .89
Sensitivity: A' = .90

Results for a small sample size of 6

✓ ***Bias***: The Usability Scale Score appears unbiased

- β for a biased system can vary widely; e.g., from 0.2 to 9.0
- β = 1.0 indicates a perfectly unbiased detection system
- Obtained **β'** values of .94 and .89 indicate scale scores are unbiased

✓ ***Sensitivity***: The Usability Scale Score appears reasonably sensitive

- ***A*** varies from 0.5 to 1.0
- ***A*** = 0.5 indicates random guessing, *A* = 1.0 is perfect accuracy
- Obtained ***A'*** values of .89 and .90 are okay (but not great)

What These Results Mean

- The Usability Scale Score leads to relatively unbiased decisions
- Decisions are fairly accurate with sample sizes as small as 6 users
- But false alarm (and miss) rates are higher than we would like
- The Usability Scale Score should perform even better with larger sample sizes; for example, 15 – 30 users