

# Experimental Approaches in Computer Science

Dror Feitelson  
Hebrew University

Lecture 10 – Experiments with People

- Computer systems are developed by people
- Computer systems are maintained by people
- Computer systems are used by people
- These are not the same people
- It is hard to anticipate what other people will think and do

People are different  
from each other,  
and they are different  
from you!

Need to experiment to find out

## Areas of application

- Empirical software engineering
  - How do people design systems?
  - What are generally good procedures?
- Usability testing in HCI
  - How do people use systems?
  - What are generally good interface design guidelines?
- User perception of system performance
  - What do users care about?

# Recurring questions

- Which processes and techniques work best
  - Example: testing vs. code inspection
  - Example: detailed design vs. extreme programming
  - Example: wide menus vs. deep menus
- Variation between experienced and novice programmers / users
  - How to make it transparent for newbies
  - How to make it efficient for experienced users
  - Can both use the same mechanisms?

# Experimental techniques

- Observation and data analysis
  - See how users behave on their own
- Controlled experiments
  - See how users perform predefined tasks
  - See how user behavior changes when a specific system parameter is changed
- Interviews and surveys
  - Try to understand why users behave the way they do

# Observing User Behavior

Question: what do users care about?

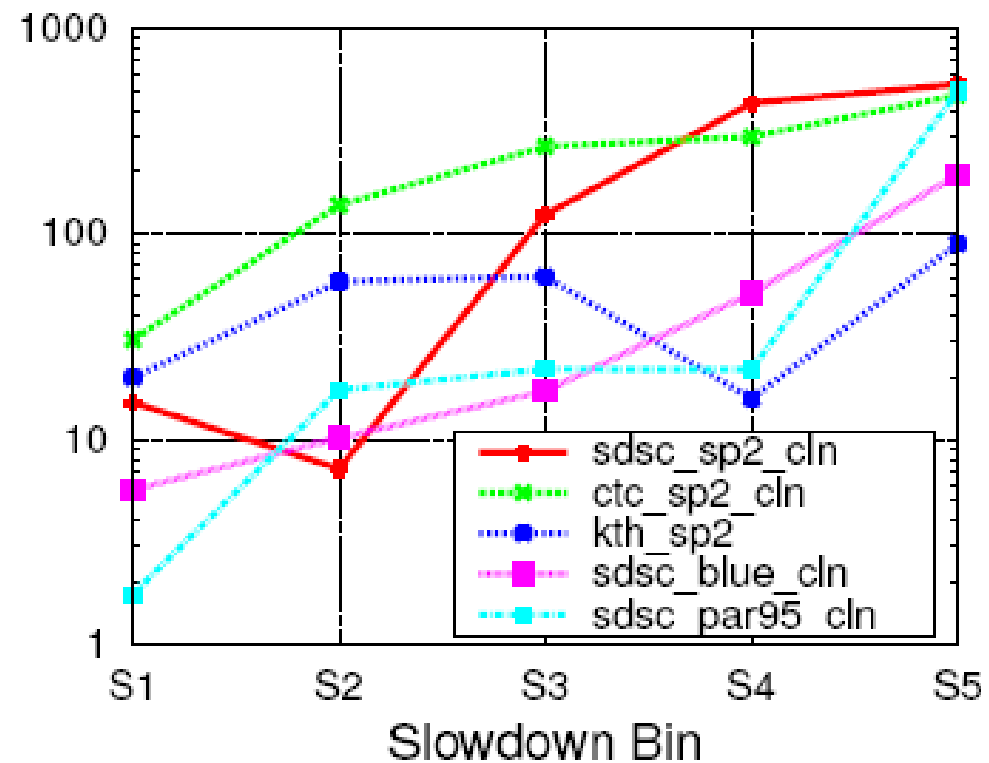
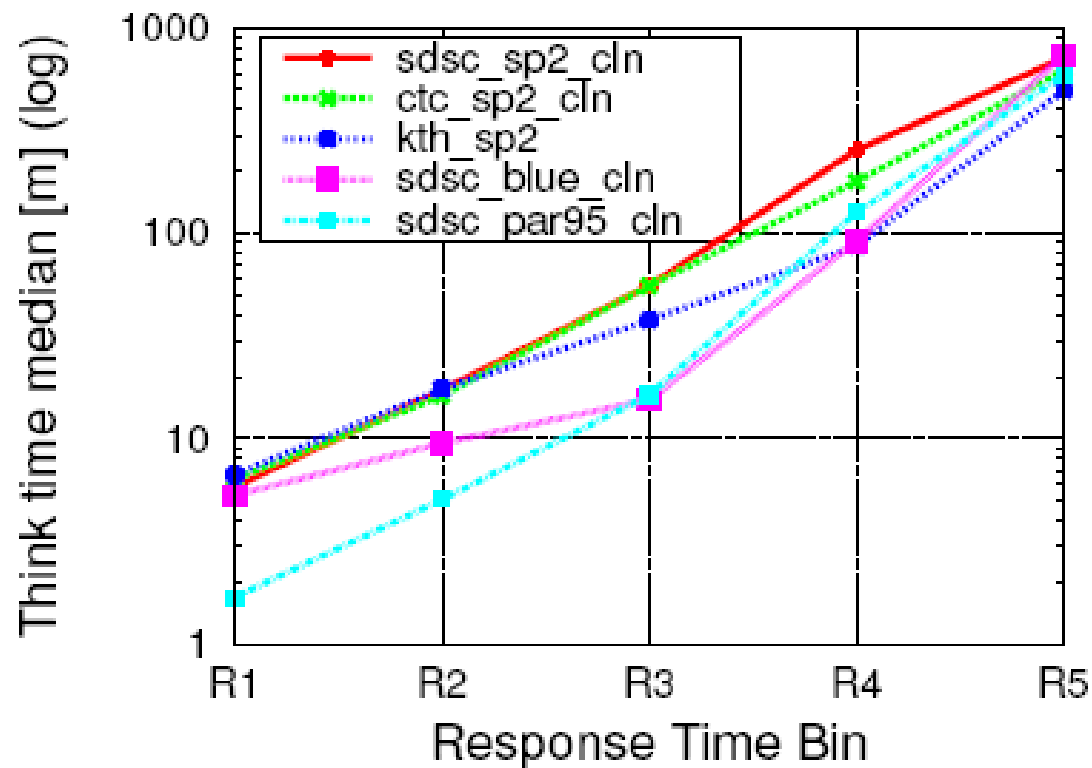
- Response time
  - This is how much time the user has to wait
- Slowdown
  - Response time normalized by actual execution time
  - This reflects the deviation from what could be expected in advance
- Optimizing for one may be different from optimizing for the other

Idea: look at user behavior

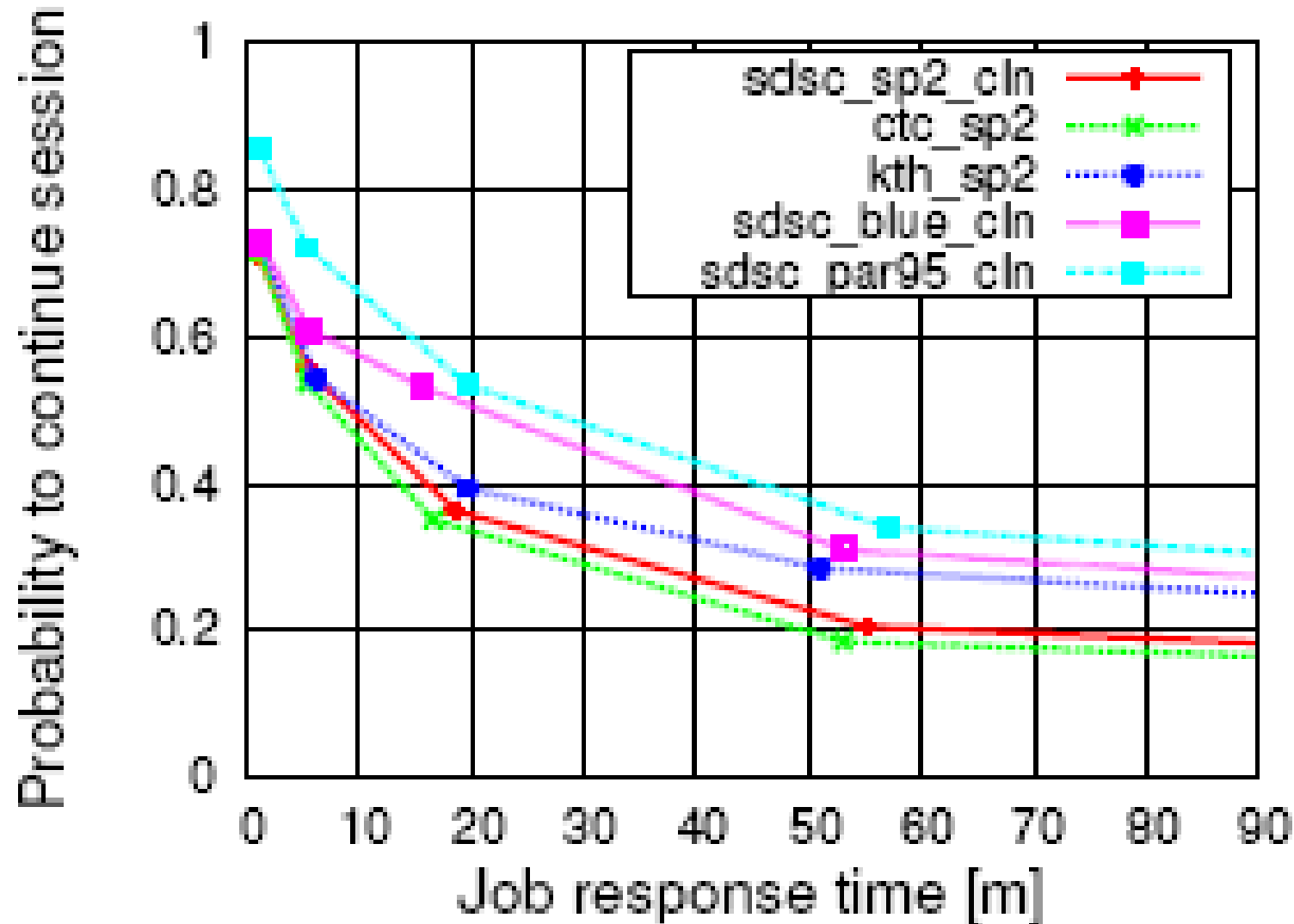
- Use logs of system activity
- Look at the activity of each user separately
- For each user job, tabulate its performance
  - This is expected to affect subsequent user actions
- Then tabulate the actual behavior
  - For example, the think time till the next job is submitted
- Correlate the performance with the behavior to see the effect



- Actual data is from parallel jobs on supercomputers
- Results: response time has consistent effect on think times, slowdown does not



Alternative view: with higher response times, probability to continue interactive session is reduced (measured as think time < 20 min.)



**Contextual inquiry:** detailed observation of a small number of people during their normal work

- What are the real problems
- What are the real needs
- Where can you bring real value
- A deep level of requirements elicitation
  - Users often don't know to articulate what they need
  - What they want is not always what they need
  - What they need may not be computer related (e.g. solve the problem of coffee always being spilled on the printout)

## Actually doing it

- You are the apprentice learning from the user who is the master (even if you know more)
  - Not interviewer, guest, or reviewer
- Take notes about actions done, their sequence, and tools/artifacts used
- Understand what the users did and why
  - Model the flow of information
  - Model the sequence of operations
  - Model the environment and interactions with it

# Controlled Experiments with Users

## Steps in a user study:

- 0) Define the system's goals – what services does it provide?
- 1) Create a set of tasks that are performed in order to meet these goals
- 2) Get people who are representative of the system's users
- 3) Watch them try to perform the tasks

applicable to customers of a web site as well as developers of a new application

## Why does this thing exist?

- **User perspective:** What are the specific goals of using this system, that differentiate it from other systems?
- **System perspective:** What are the key features of the system that make it valuable?

## Write a short description of what this is all about

- A couple of sentences
- Will be used as introduction to testers

## Selecting tasks for a usability study

- Too many, so can't test all
- Make a list of tasks and rate by importance to the product on a scale of 1 to 5
- Now rate by the degree of doubt that the designers have about them, again 1 to 5
- Multiply the two ratings and sort
- Test the top-ranking tasks: those that are important and most require user input



## Defining the tasks

- Define the **goals**, not the **procedure**
  - **Want to find out what procedure users will use**
- Be specific about exactly what's needed
- Create a reasonable sequence
- Don't use words that appear in the user interface
- Together they shouldn't take too much time
  - Estimate how long it will take you (an expert who knows the system)
  - Multiply by 3 to 10

# Getting people for the study

- **Recruiting:** finding some people
  - Based on general demographics and mix
  - Age, gender, income level, computer usage level
- **Screening:** finding the right people
  - Filter out those that match the demographics but are probably not useful
  - Interested (but not predisposed) in the system, maybe use similar system
  - Articulate about expressing what they think
  - Available on the planned test dates
  - Not working in the industry

# Conducting a user session

- Explain that the user is helping to test the system, as opposed to the system testing the user
  - There are no wrong answers
  - If you don't understand it is OK, and in fact our main goal is to know about it
  - Analogous to people hired to watch a show pilot
- The user should say out loud what he is trying to do and why
  - Again, don't be ashamed
  - Best to videotape the whole process
  - The study organizer is in the background only

# Experimental aspects

- Measure the time to perform different tasks
  - Average of each task across users
  - Average of each user across tasks
- Record on video for later analysis
  - Screenshots
  - User behavior and expressions

# Experimental plan

- Better to have multiple small studies than one huge study
- Number of subjects can be as low as 5-6
  - Enough to get feel for results, not necessarily good statistics
- Conduct a pilot session
  - Find out suitability for different user demographics
  - Verify that tasks are reasonable
  - Verify that description of system and tasks are understandable

# Conducting Interviews

# Important issues

- Population being interviewed
  - Question of sampling bias: you want generally representative users
  - Question of required sample size
- Phrasing of the questions
  - Questions should be neutral so as not to affect results
  - Question order is also important
  - Imperative to pre-test the questions on a small sample to detect and correct problems
- Statistical analysis of results

- Unstructured interview
  - Completely free exchange
  - Used as an exploratory tool in initial stages of a study, when the researcher doesn't know much yet
- Semi-structured interview
  - basically follow a pre-defined outline of questions
  - allow user to expand on various topics
  - also on-line questionnaire where questions depend on previous answers
- Structured interview
  - Filling out a pre-defined questionnaire



## Questionnaire structure

- Title
- Short introduction – what is this about
- Demographics questions – who is answering
- Start with the easy questions
- Leave sensitive questions to the end

- Questions types:
  - Multiple choice or a scale
  - Numeric (how many times a day do you sneeze?)
  - Open text (what do you do in the morning?)
- When giving choices, always include "N/A", "other", etc.
- Provide text explanations in addition to scale
- Desirable scale is debatable
  - Number of points is 5 to 10
  - Even forces a decision, odd allows an undecided middle response

Characters on the computer screen are:

1	2	3	4	5	6	7	8	9	N/A
hard to read					easy to read				

Image of characters are:

1	2	3	4	5	6	7	8	9	N/A
fuzzy					sharp				

Character shapes (fonts) are:

1	2	3	4	5	6	7	8	9	N/A
barely legible					very legible				