Human-Computer Interaction

[Preece Chap1-2; Hix Chap1; Nielsen Chap2]

What is Human-Computer Interaction?

A brief history of computing.
The first computers were extremely large, complex machines that were used exclusively by specialists. Their user interface was switches, then punch cards. Everyone was an expert.
Computers were very batch oriented (non-interactive).
As the transition occurred from mainframes to the personal computer, now specialists weren't the only people using computers.
Interfaces had to account for the beginner and intermittent user as well as the expert.
Now, computers are in many homes, being used by non-technical people.
Xerox Star — Apple Macintosh — Windows 95
WYSIWYG Interfaces (GUI)

Computers do certain things very well — sorting, searching, repetitive tasks, etc.
People do certain things very well — subjective tasks, pattern recognition, etc.
These things are most often complimentary.
Synergy — If we could design a system that makes optimal use of the innate skills of the human operator and the machine, it would far outweigh the effectiveness of either part individually.
We are interested in humans within the context of computing systems.

Software designers must account for who their users are!
But do they?
It is estimated that on average 48% of application code is dedicated to the interface. Is 48% of the effort dedicated to its design?
Functionality isn't there if your users can't find it.

Components of HCI
"an input language for the user, an output language for the machine, and a protocol for interaction" (Chi, 1985).
See Figure 1.
See Figure 2.

For large-scale products, the design and implementation team is extremely large.
Managing this team and maintaining focus on the product from the user's (consumer's) perspective is a non-trivial task.

The challenge of HCI
As technology changes, the relationship between users and computers changes in unpredictable ways.
What is an optimal interface? How would you know if you saw one?
We want usable, safe, systems — also inexpensive, low maintenance, low learning curve, high productivity, etc.
Example: The NYNEX operator analysis.
Why don't the best systems do the best in the market? Incremental improvements, lower learning curves.
The Limits of Innovation, (Lewis, 1996).

What is usability?

Making systems that are easy to learn and easy to use.
Less is more: adding marginally useful functionality usually complicates a system and lowers usability.

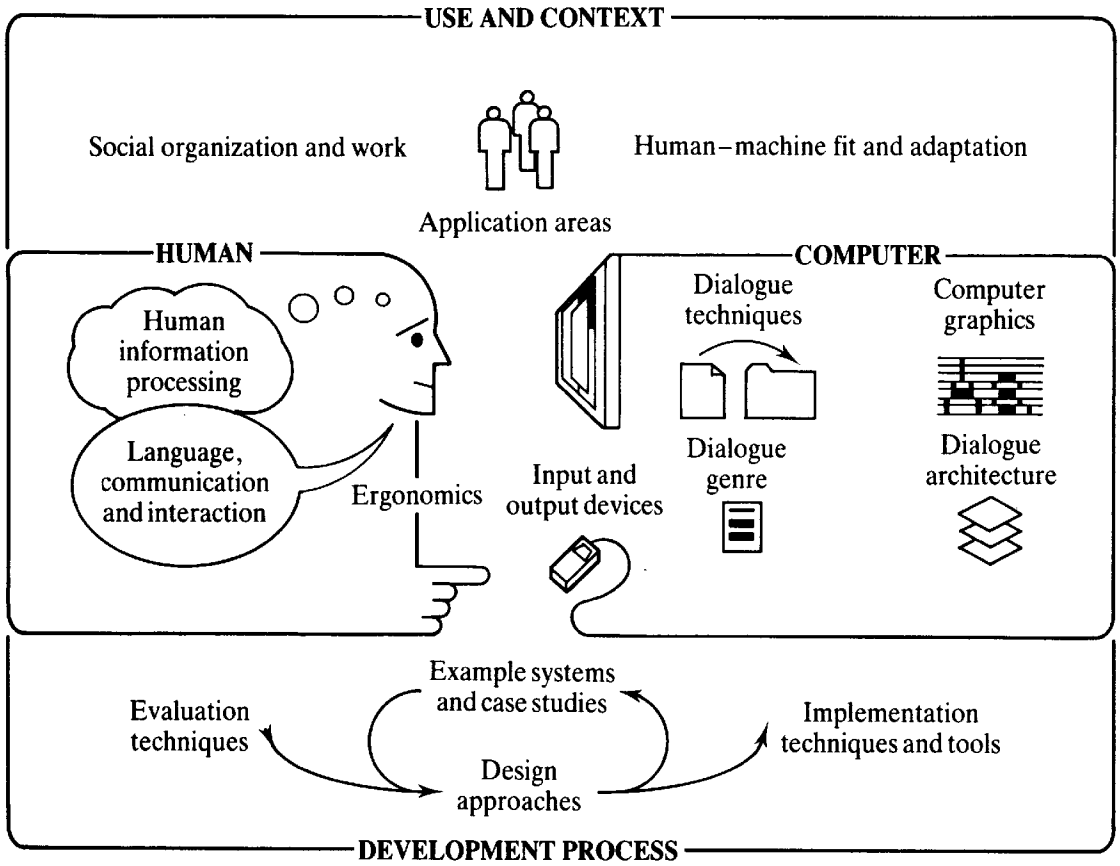

Figure 1.

To the user, the interface IS the product. Usability IS the measure of its quality.
Definition of usability (Nielsen, 1993):

- **Learnability**: The system should be easy to learn so that the user can rapidly start getting some work done with the system.
- **Efficiency**: The system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible.
- **Memorability**: The system should be easy to remember, so that the casual user is able to return to the system after some period of not having used it, without having to learn everything all over again.
- **Errors**: The system should have a low error rate, so that users make fewer errors during the use of the system, and so that if they do make errors they can easily recover from them. Further, catastrophic errors must never occur.
- **Satisfaction**: The system should be pleasant to use, so that users are subjectively satisfied when using it; they like it.

See Figure 3.
The ease-of-use versus power-use trade-off. Window OS versus Command line OS. (See Efficiency).
See Figure 4.

Why is usability important?

Costs of unusability...
To end-users:

Too many systems are only technically successful,

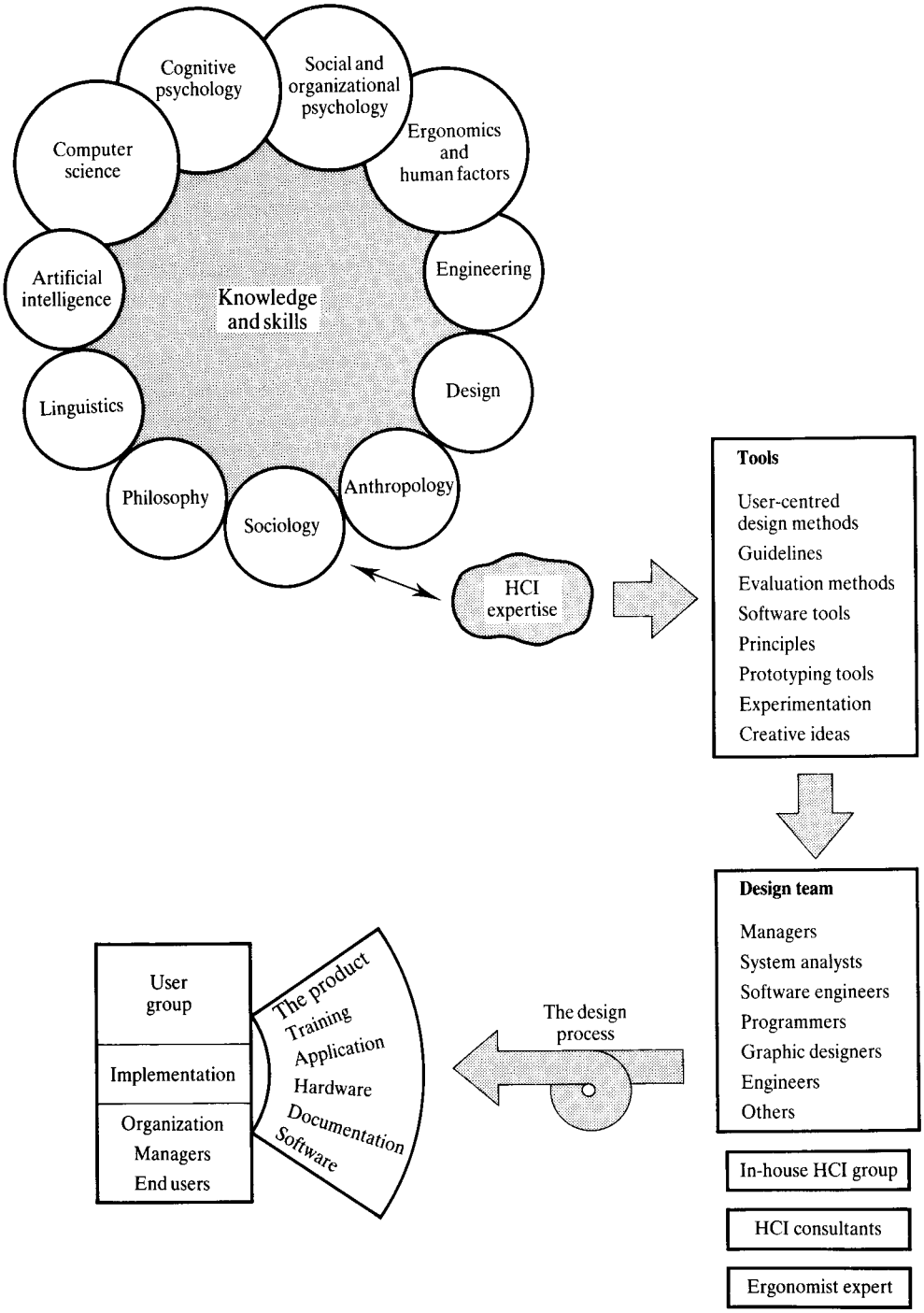
Cognitive psychology
Social and organizational psychology
Ergonomics and human factors
Computer science
Engineering
Artificial intelligence
Knowledge and skills
Design
Linguistics
Philosophy
Sociology
Anthropology
HCI expertise
Tools
User-centred design methods
Guidelines
Evaluation methods
Software tools
Principles
Prototyping tools
Experimentation
Creative ideas
Design team
Managers
System analysts
Software engineers
Programmers
Graphic designers
Engineers
Others
In-house HCI group
HCI consultants
Ergonomist expert
The design process
User group
Implementation
Organization
Managers
End users
The product
Training
Application
Hardware
Documentation
Software

Figure 2.

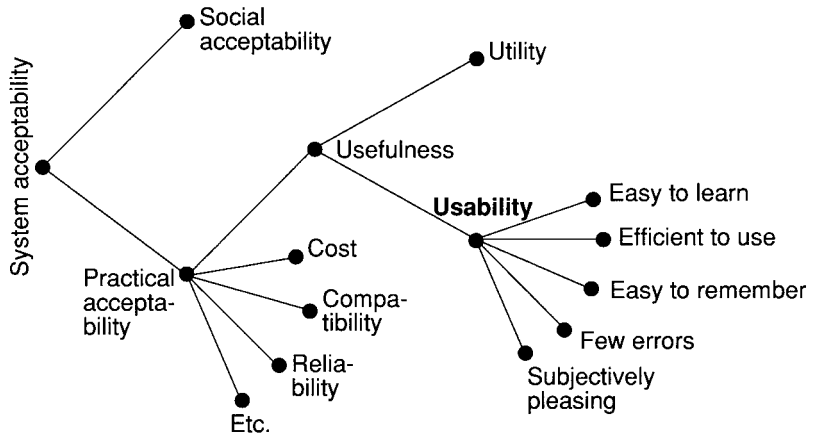

Figure 3.

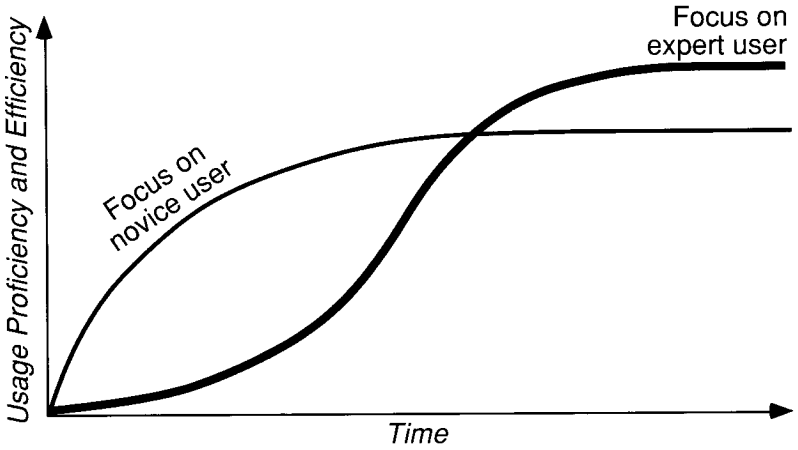

Figure 4.

Because their users find them to be:

Confusing to comprehend
Inconsistent
Requiring excessive training
Difficult to retain, requiring retraining
Error prone
Discouraging to explore to full advantage
etc., etc., etc.

"Computers and bureaucracies are really well-matched, Both operate only at their own convenience and neither shows any awareness of their intended purpose."
What is that purpose? — To serve the user. To maximize performance of the man-machine pair, not one or the other. The computer is not my friend it's my servant.

To developers:

Lack of a systematic approach to usability -

Causes false starts
Extends development schedules
Consumes maintenance budget with "usability patches"
Increases complexity of code

Lowers customer satisfaction

“It’s easy to make things hard. It’s hard to make things easy.” (Chapanis, 1982)
User Friendly != Programmer Friendly
A GUI does not necessarily mean a system is usable.
“Learning to use a computer is like learning to use a parachute — if a person fails on the first try, odds are he won’t try again.”

What is the value in usability engineering?

- Avoid catastrophe (3-Mile island)
- Reduce errors
- Keep employees happy
- Decrease training time
- Increase throughput
- Require fewer “help” calls
- Attract new customers