interaction design history in a teeny little

marc rettig

marcrettig.com

presented at
carnegie
mellon
university

2 april 2004

mrettig@well.com

history

fine print
this presentation was originally prepared in 2003 for a lecture in Melissa Cicozi's undergrad Design History class at Carnegie Mellon University. This slight revision was presented in 2004, in Jodi Forlizzi's Graduate Interaction Design Seminar at CMU.

this isn't a history of computing technology.
This isn't a history of interfaces.
It is an overview of the history of designing for interaction.

standard disclaimers apply: this was not carefully fact-checked, and it very much represents my personal view of what has been and is now important. Caveat emptor, and bon appetit.

Consider these planes (an ancient tool): their designers sought fitness for use, ease of use, good control, long-lasting materials, a good feel in the hand, efficiency of operation, precise adjustment. In use over time, these tools come to be loved by their owners.

operate the machine

pre-computer

Before computers, there wasn’t “interaction design.” But most of the qualities we seek have been valued through the ages.

- useful
- usable
- desirable
- affordable for the right people
- appropriately complex
- appropriately styled
- appropriately transparent in function and use
- appropriately adaptable, extensible, malleable
- overall, having “good fit” with people, context, activity, result

When programmable, interactive machines first appeared, the creators of their controls, their interfaces, emphasized the goal of “operating the machine.”

operate the machine

back in the day

- design was *engineering* design:
  make faster, bigger machines, expose their guts through controls
- people adapt to the machines
- people speak the language of the machines
- elaborate efforts to prepare problems for the machines
- no designers involved, but lots of clever engineers – emergence of a new set of skills, new disciplines
- in the meantime, the field of human factors is blooming, along with things like “aviation psychology”

operate the machine

characteristic statement of the time

people are seen as components in a system of production

The Five Elements of System Design

personnel selection

personnel training

machine design

job design

environmental design

a current statement of the goal of “human factors”

my summary

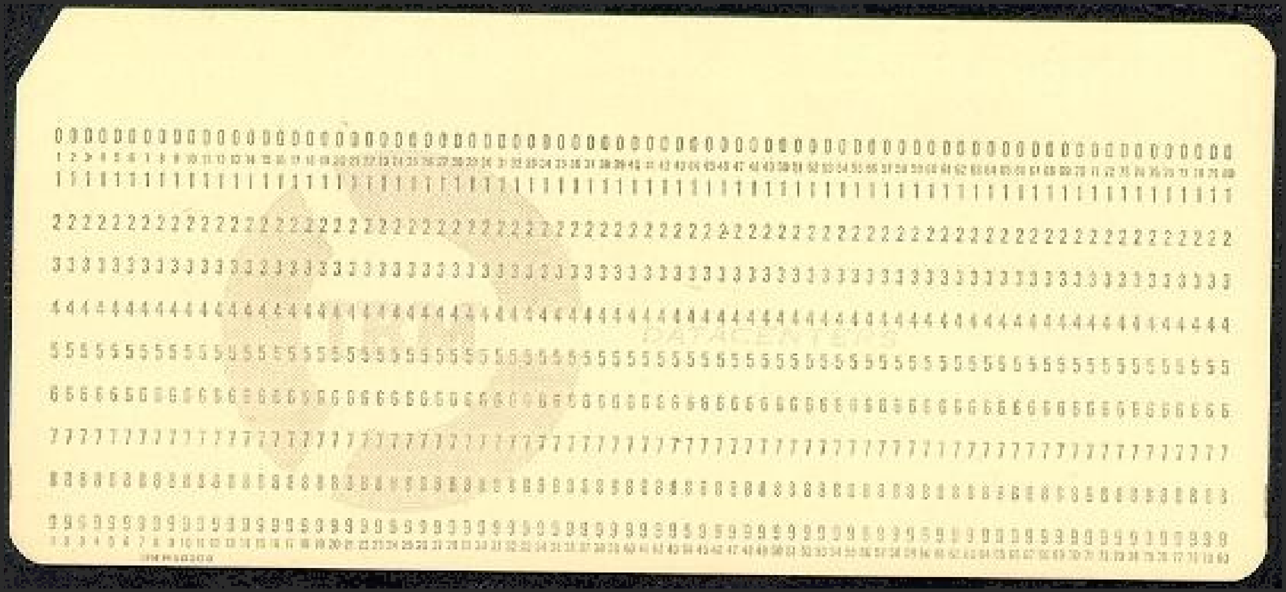
“minimize the damage and inconvenience”

Good Designs:

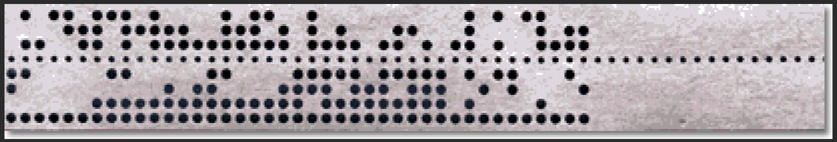
- design against misuse, unintended uses, and abuses
- design for all sizes, shapes, attitudes and personalities people

input and output: people adapt to the machines

punch card,
80 columns, to hold 80 characters or numbers



paper tape, also encoding characters with holes.



For fun, go make images of punch cards that say anything you want:
http://www.facade.com/legacy/punchcard

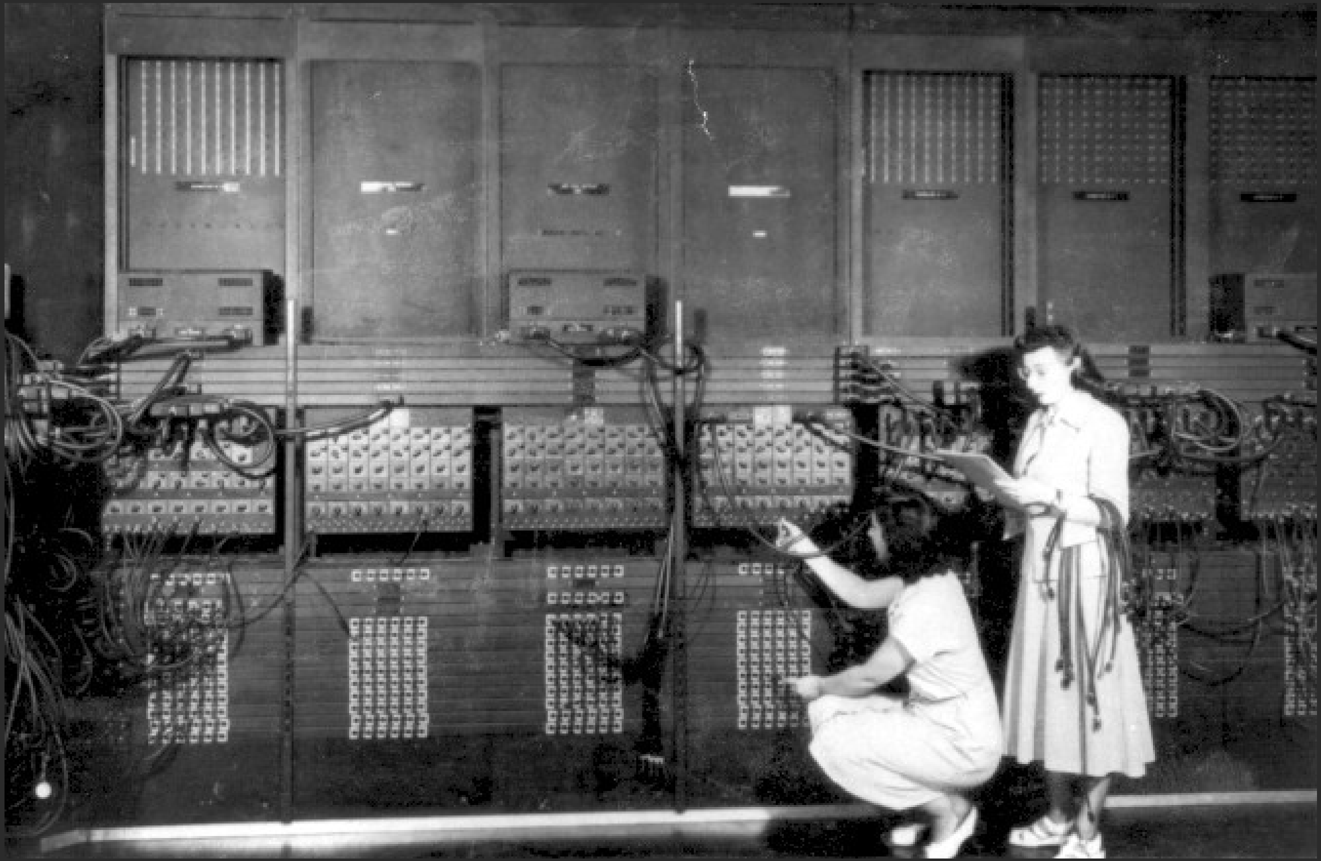
wiring the ENIAC with a new program

ENIAC
1946
Mauchly and Eckert

stats:
3,000 cubic feet
30 tons
18,000 vacuum tubes
70,000 resistors
170 kilowatt power req.
~1 kilobit memory

approximate processing power of today’s singing birthday card

but **not** a stored-program device



Great description here: www.computinghistorymuseum.org/teaching/lectures/pptlectures/7b-eniac.ppt

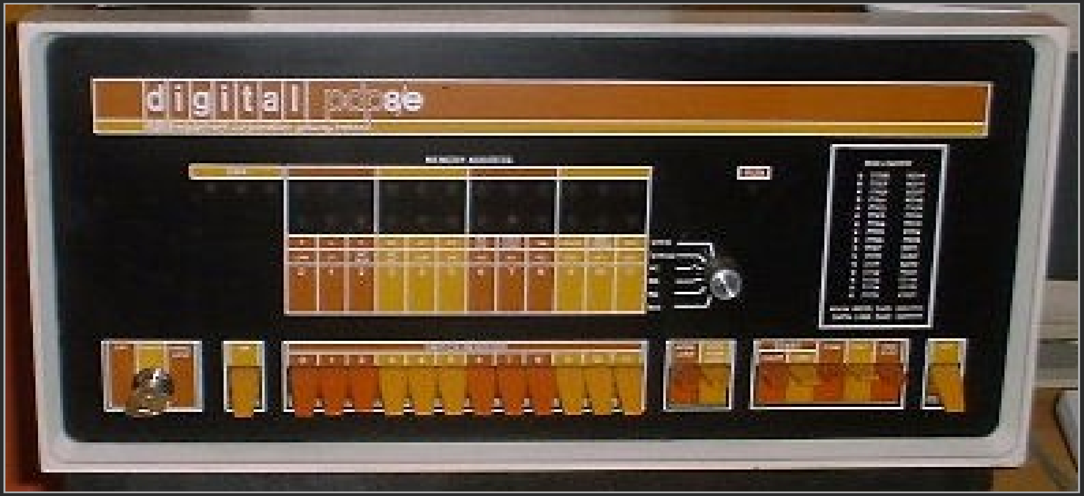
front panel switches

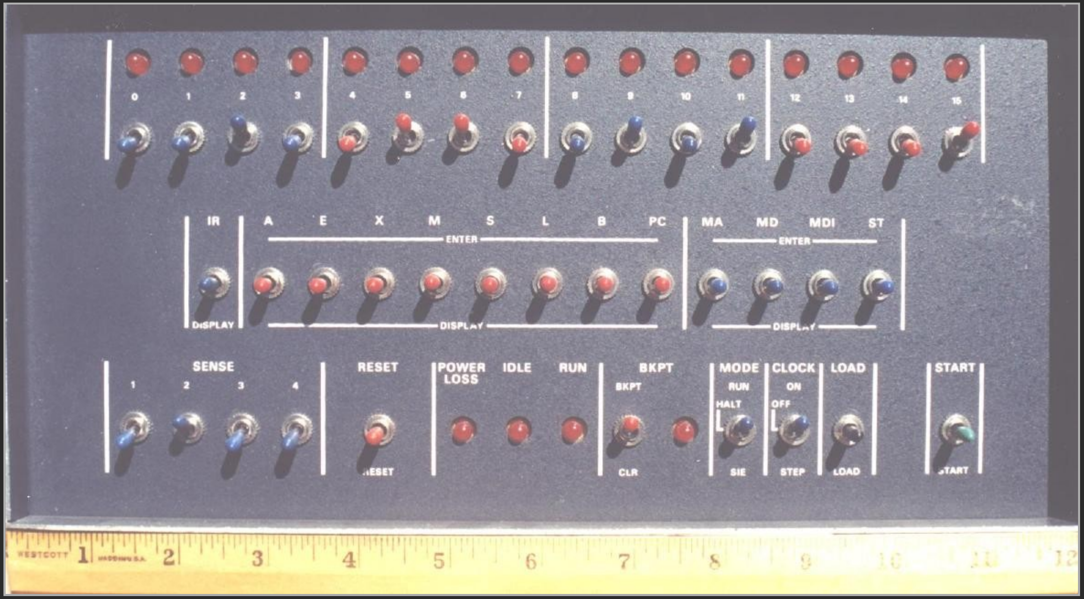
DEC PDP-8

TI 980

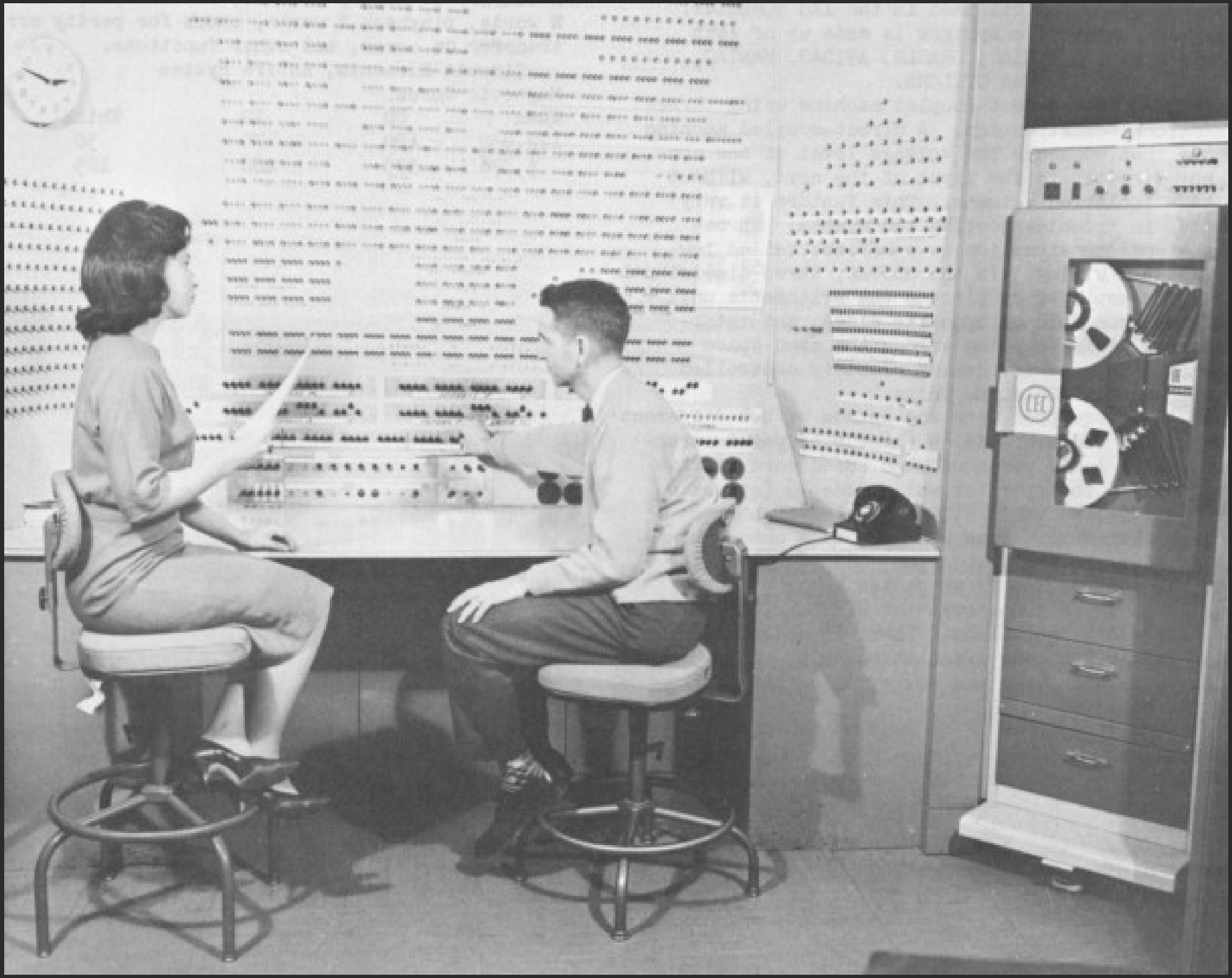
1960’s

The internal architecture of the machine is exposed in the controls. You can see that the PDP-8 is an octal computer, with its switches in three-bit configurations (it takes three bits to count from 0 to 7, for a total of 8 numbers. Base 8. Octal. Get it?). The TI 980 is a hexadecimal machine, with switches in groups of four. Using the switches, you program the machine one word at a time (a word being, say, two hexadecimal bytes for the TI).





configure switches, run batch, output to tape



batch processing: feed it cards, wait while it runs

What you used to do
punch a deck of cards; take the cards to a little window, hand them to the operator; she puts them in line with everyone else’s jobs; when it’s your turn she puts your cards in the hopper and pushes “RUN”; your program works or it doesn’t; an hour or twelve later, you pick up your cards and (hopefully) printout at the same little window.

What you do now
double-click an icon, see what happens immediately.

preparing punch cards

An important by-product: confetti. All the chaff from all those cards was just *great* to throw around the dorm.

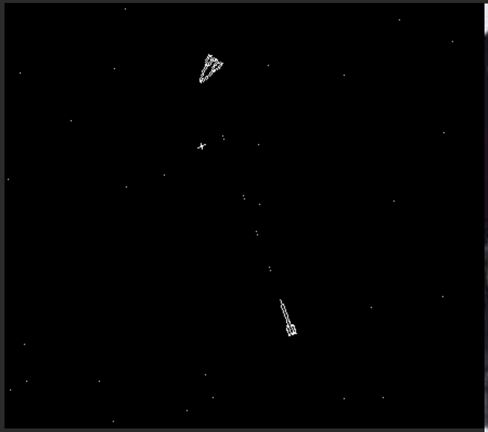
preparing punch cards

Each key press punches holes, so there's no "erase." Fixing a mistake almost always required ejecting the card and starting it over.

In a pinch – say you *really* needed to fix a card and the punch was down – a clever operator might know enough about the card encoding to close some holes with tape and open others with a knife.

So on the one hand, we were adapting to the machines. On the other hand, the workings of the machines were exposed, right out where we could get to them.

online processing

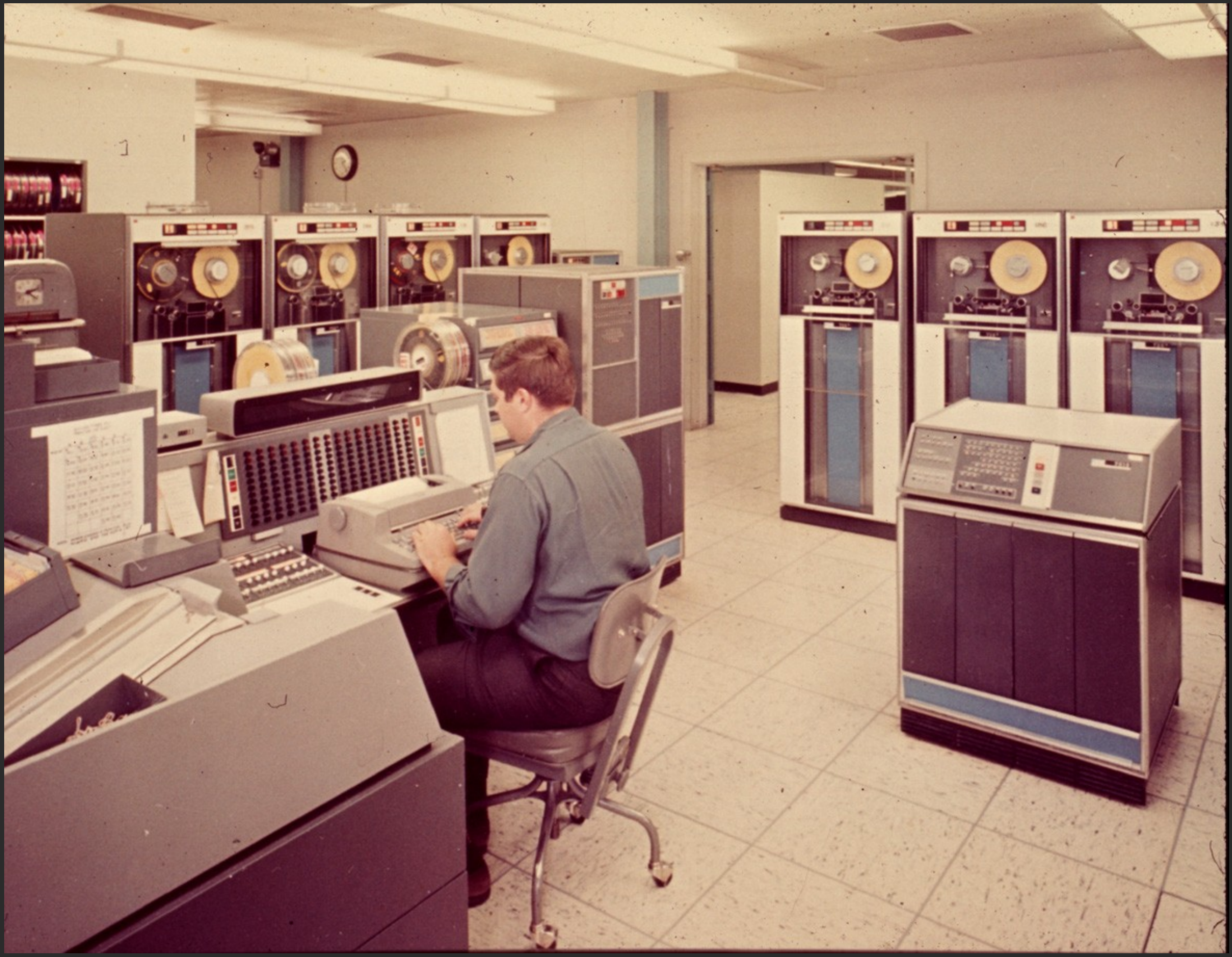


Spacewar
Steve Russel
1962

Wanna try it? lcs.www.media.mit.edu/groups/el/projects/spacewar/

operator console

IBM System 360
1960’s



remote terminals attached to the S/360

IBM 3270
1970’s

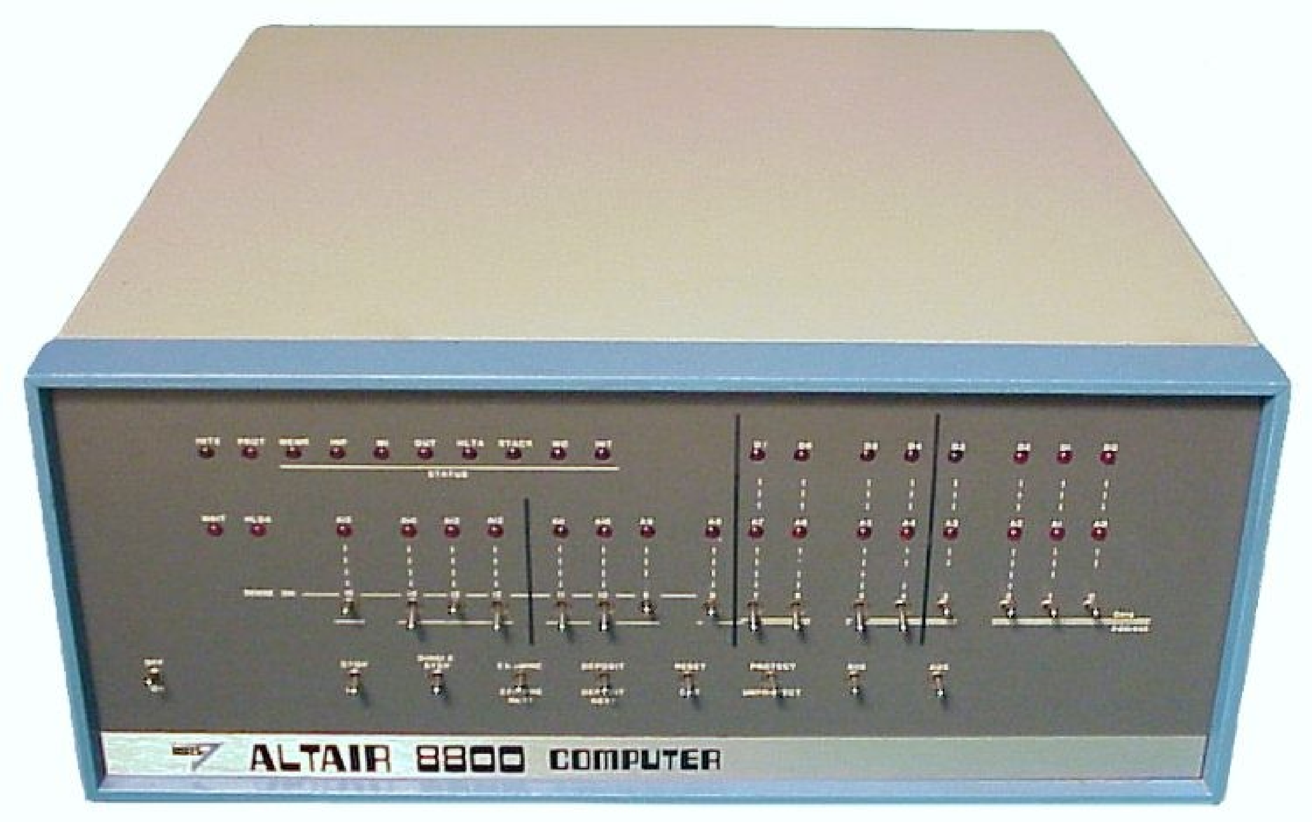
80 columns x 24 lines
a.k.a., “80 cards”

Don’t laugh. These are very hip boys.

at home, it’s still the switches – but what to *do* with it?

MITS Altair 8800
1975

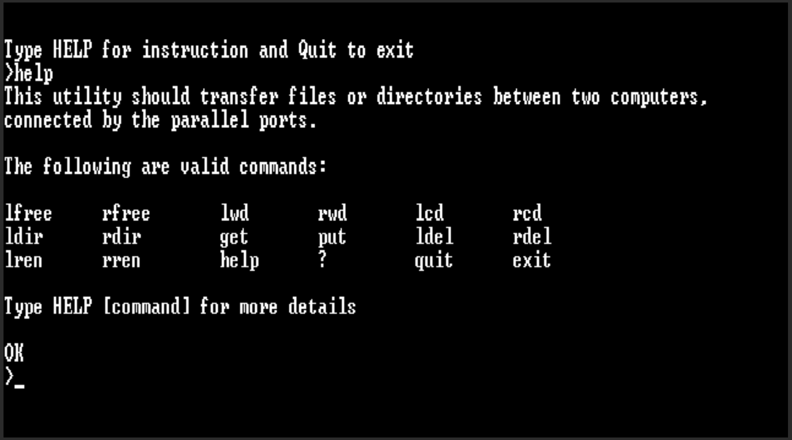
One of the first commercially available home computers. You ordered it. You built it. You operated it through front panel switches.



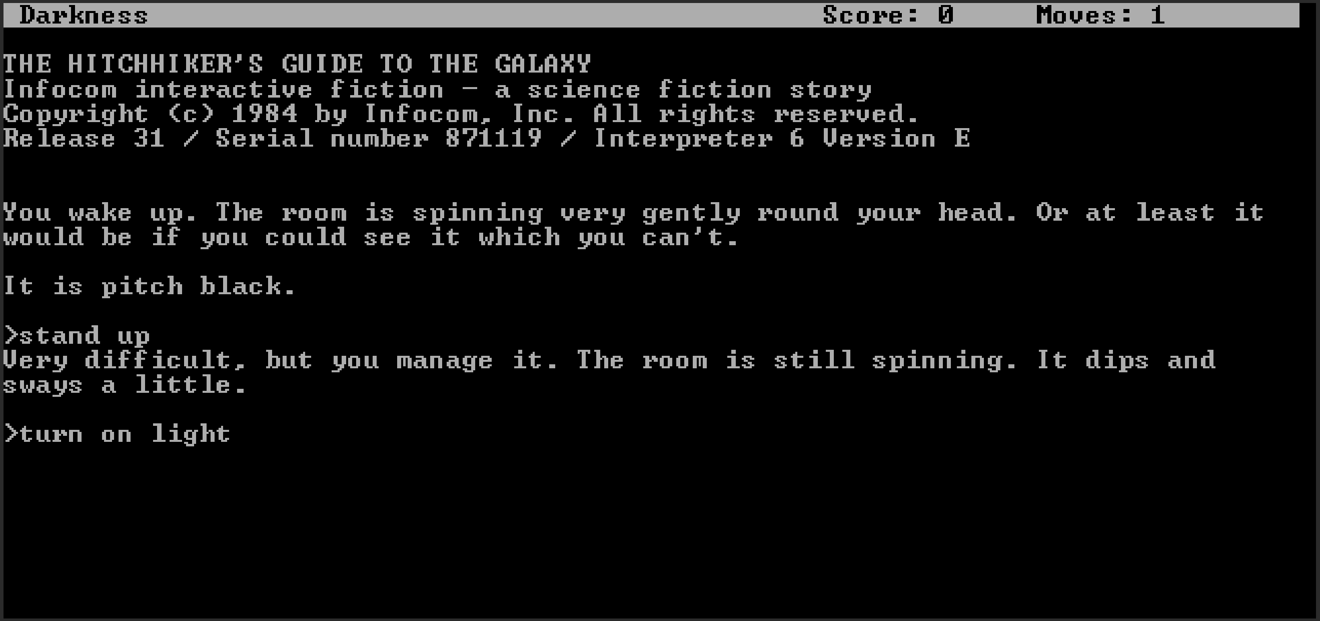
command line interfaces

Very efficient once you learned them. At least the good ones were / are. Of course they still exist, and have finally come to your Macintosh!

Still, the emphasis is very much “operate the machine.”



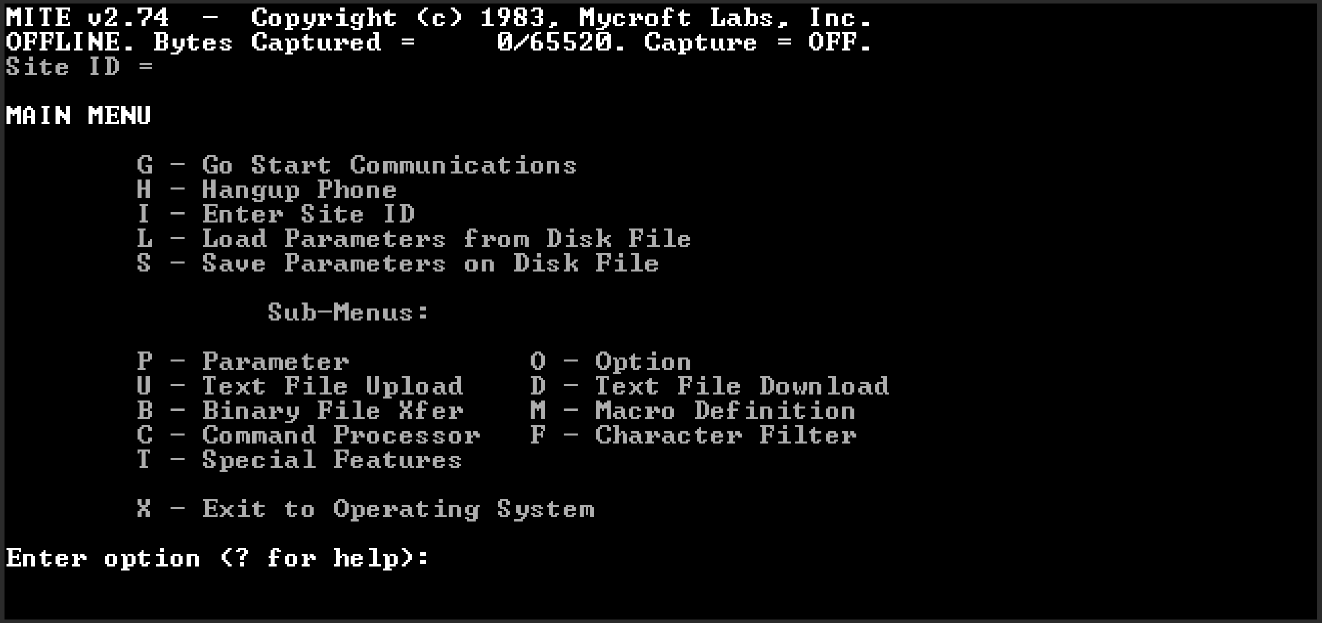
```
Type HELP for instruction and Quit to exit
>help
This utility should transfer files or directories between two computers,
connected by the parallel ports.

The following are valid commands:

lfree    rfree     lwd      rwd      lcd      rcd
ldir     rdir      get      put      ldel     rdel
lren     rren      help     ?        quit     exit

Type HELP [command] for more details

OK
>_
```



```
 Darkness                                          Score: 0     Moves: 1

THE HITCHHIKER'S GUIDE TO THE GALAXY
Infocom interactive fiction - a science fiction story
Copyright (c) 1984 by Infocom, Inc. All rights reserved.
Release 31 / Serial number 871119 / Interpreter 6 Version E


You wake up. The room is spinning very gently round your head. Or at least it
would be if you could see it which you can't.

It is pitch black.

>stand up
Very difficult, but you manage it. The room is still spinning. It dips and
sways a little.

>turn on light
```

“user friendliness”

“User Friendly” was a *huge* buzz phrase for years. Early on, it meant things like providing clear help and easy to remember command names. A great and still relevant book from the time: Paul Heckel’s *Elements of Friendly Software Design*. Still available from Amazon.

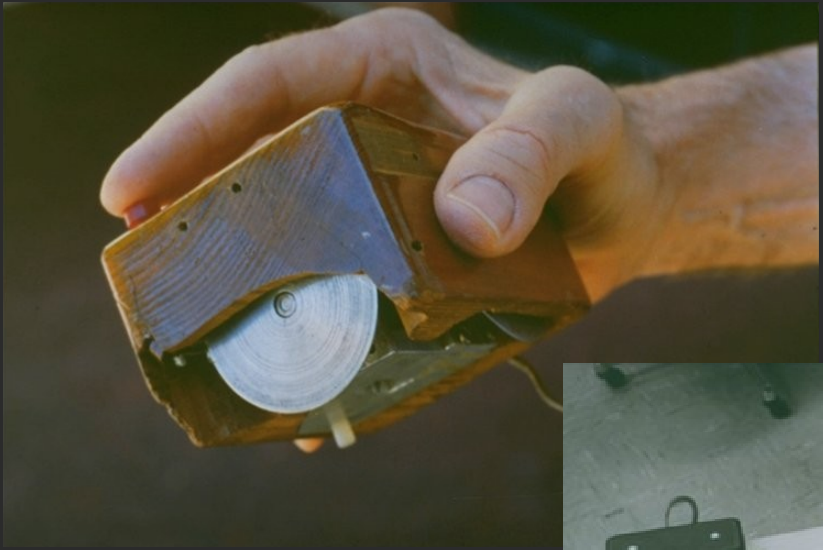


```
MITE v2.74  -  Copyright (c) 1983, Mycroft Labs, Inc.
OFFLINE. Bytes Captured =     0/65520. Capture = OFF.
Site ID =

MAIN MENU

        G - Go Start Communications
        H - Hangup Phone
        I - Enter Site ID
        L - Load Parameters from Disk File
        S - Save Parameters on Disk File

                Sub-Menus:

        P - Parameter               O - Option
        U - Text File Upload        D - Text File Download
        B - Binary File Xfer        M - Macro Definition
        C - Command Processor       F - Character Filter
        T - Special Features

        X - Exit to Operating System

Enter option (? for help):
```

in the meantime, a few people were thinking differently

mouse
Doug Englebart
1964

A landmark event in the history of interaction design: Doug Englebart’s 1968 demo at SRI. He demonstrated
most of the ideas we associate with modern desktop computing: the mouse hypertext, objects in the interface, dynamic file linking, and even two people at different locations communicating over network audio and video. This work was done from a human-centered point of view, and the demo is required viewing. Watch it, remember it’s **40** years ago, and think about how progress is made in this field.



Wanna see the demo? sloan.stanford.edu/mousesite/1968Demo.html

“you can actually *talk* to the computer”

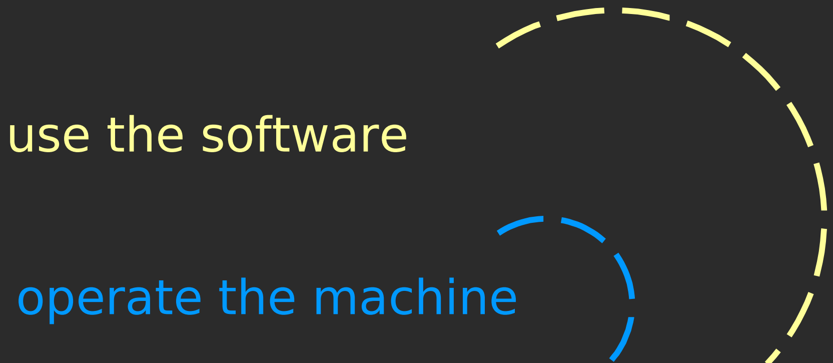
sketchpad
Ivan Sutherland
1963

CAD features in 4K RAM (?) using an oscilloscope and a light pen.

Englebart, Sutherland and others were shifting from “operating the machine” to providing people with useful tools. Englebart sought to “augment the human intellect.”

Think about a world of punch cards, then watch the video.

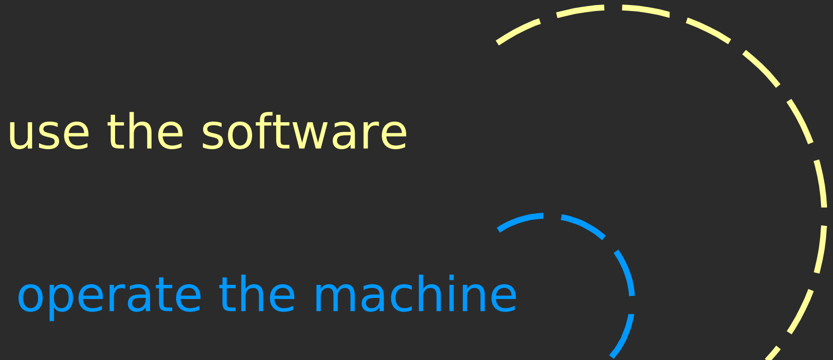
Video of Alan Kay presenting and describing this work (at Etech 2003) can be found here:
http://ftp.archive.org/movies/lisarein/oreilly/etech2003/alankay/sketchpad-ui-1963-mres.mov
(or ../sketchpad-ui-1963-mres.mov for hi-resolution). Sutherland’s demo itself is available on the ACM SIGCHI Video compilation for 1983.



use the software

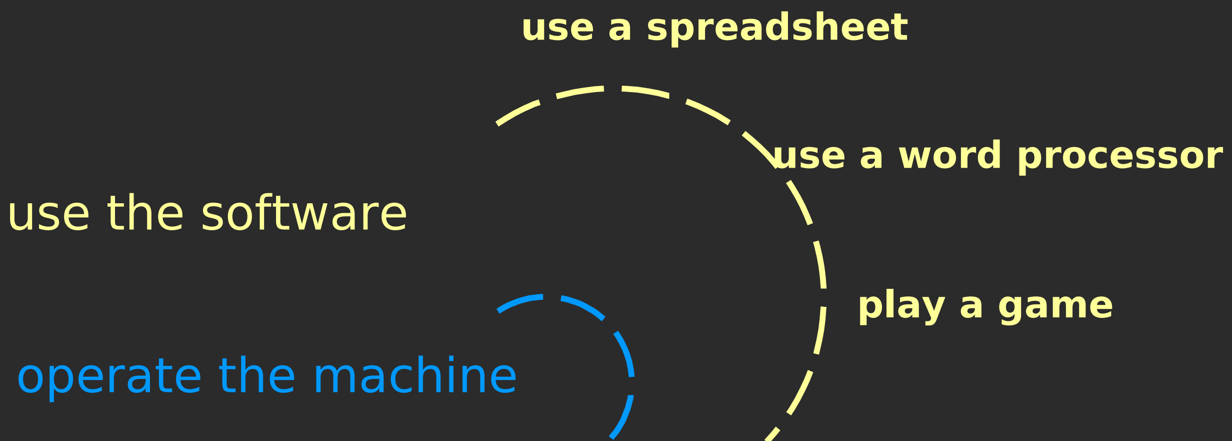
operate the machine

- shift in focus from controlling the computer to using applications and tools
- trying to make it so people have to adapt less to use the machines’ capability
- design is still done mostly by engineers, few specialists
- still mostly thought of as “computer human factors”



use the software

operate the machine



use a spreadsheet

use a word processor

use the software

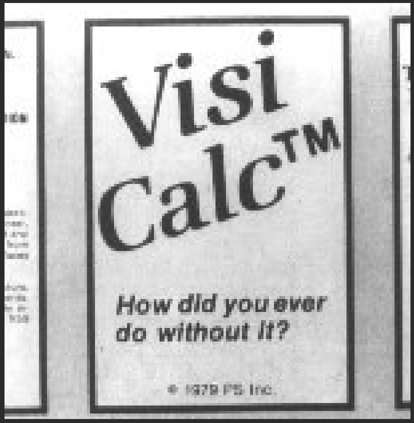
play a game

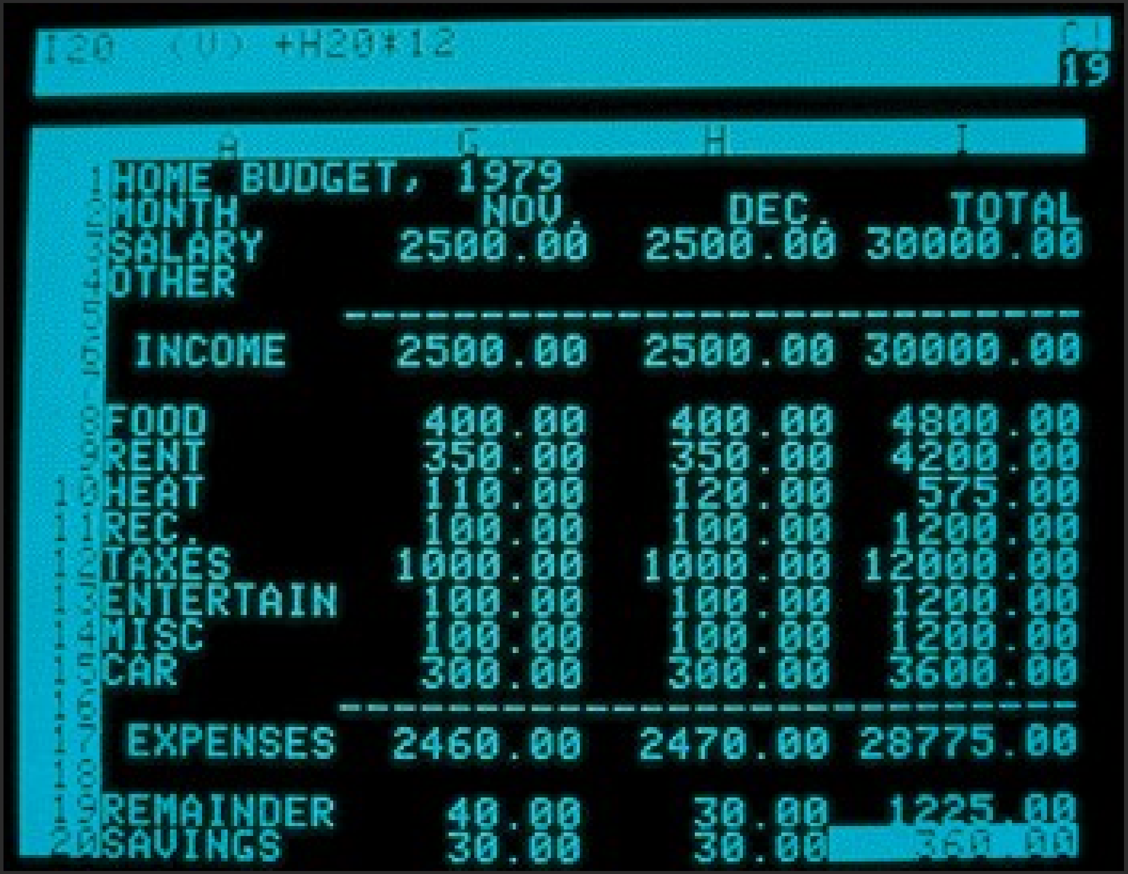
operate the machine

a tool for home and small business calculations

visicalc
Dan Bricklin
1979

Finally people had a reason to buy a home computer (specifically, an Apple II): so they could use VisiCalc, the first spreadsheet.





```
I20  (V) +H20*12                                         C!
                                                         19
        A            G            H            I
 1 HOME BUDGET, 1979
 2 MONTH           NOV.         DEC.       TOTAL
 3 SALARY       2500.00      2500.00    30000.00
 4 OTHER
 5           ----------------------------------
 6  INCOME      2500.00      2500.00    30000.00
 7
 8 FOOD          400.00       400.00     4800.00
 9 RENT          350.00       350.00     4200.00
10 HEAT          110.00       120.00      575.00
11 REC.          100.00       100.00     1200.00
12 TAXES        1000.00      1000.00    12000.00
13 ENTERTAIN     100.00       100.00     1200.00
14 MISC          100.00       100.00     1200.00
15 CAR           300.00       300.00     3600.00
16           ----------------------------------
17  EXPENSES    2460.00      2470.00    28775.00
18
19 REMAINDER      40.00        30.00     1225.00
20 SAVINGS        30.00        30.00      360.00
```

THE place to learn about Visicalc: www.bricklin.com/visicalc.htm
Download a working version!

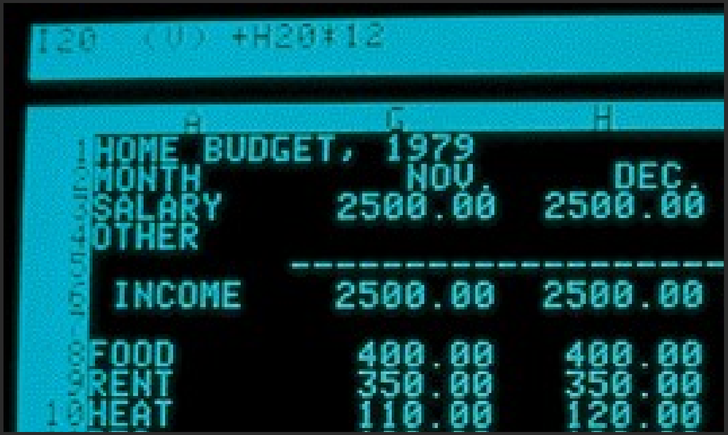
Interface and interaction ideas that survived 25 years (so far)

As Dan Bricklin points out, VisiCalc’s design has lived long:

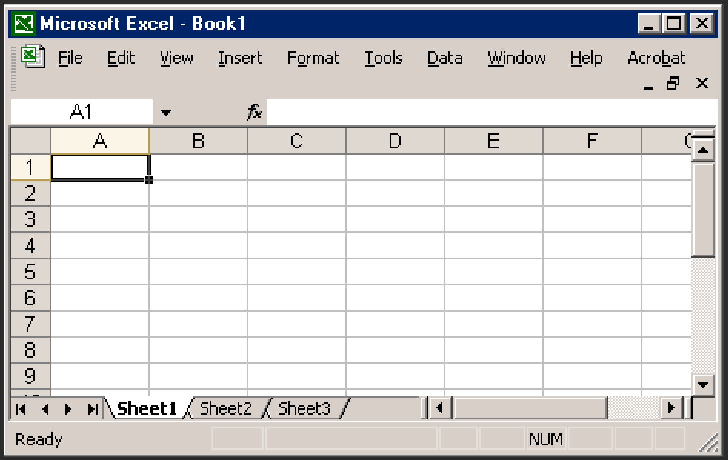
“It was interactive in a WYSIWYG way:

- Point to change a value
- Instant automatic recalculation based on formulas stored in the cells referencing other cells
- Scroll left/right/up/down
- The input, definition, formatting and output were all merged into a natural, program-by-example interface …
- Labels and formulas distinguished by first character typed
- Minimal-keystroke formula entry…. The goal here was to make it worth using the first time you needed an answer in a way that would let you benefit the next time by just changing a few values and recalculating. If the input style did not let you "teach" the computer by doing the calculation, people may not have used it.
- A1, B1, SUM(A1..A7)
- Realtime scrolling
- Numeric and text formatting
- Status and formula lines”

www.bricklin.com/visicalc.htm/firstspreadsheetquestion.htm



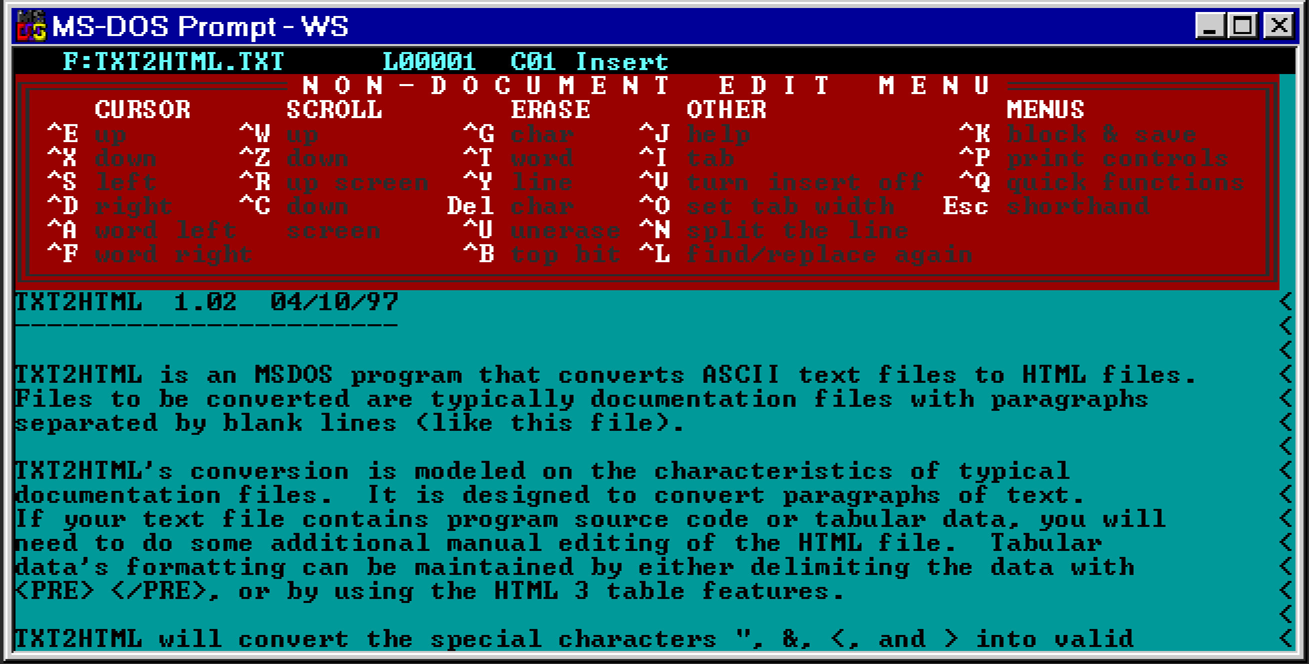
```
I20  (V) +H20*12
      A                G          H
 1 HOME BUDGET, 1979
 2 MONTH              NOV.       DEC.
 3 SALARY          2500.00    2500.00
 4 OTHER
                 ------------------
 6  INCOME         2500.00    2500.00

 8 FOOD             400.00     400.00
 9 RENT             350.00     350.00
10 HEAT             110.00     120.00
```



a tool for writing

wordstar
Seymour Rubenstein &
John Barnaby
1979

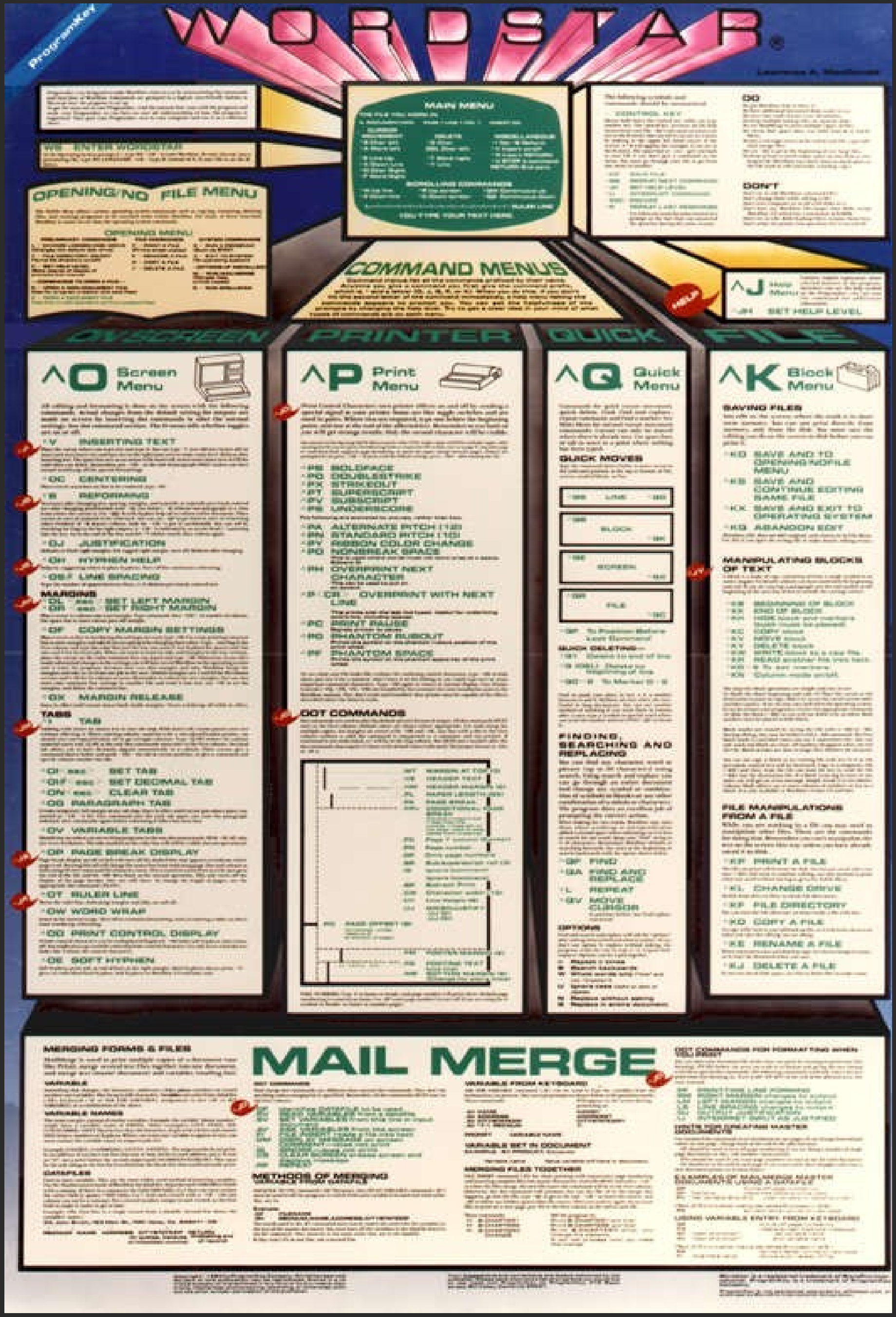
WordStar had a very complicated interface, but once you invested the time to learn it, it was very powerful. Now there was another reason to buy a home computer: to create, format, store, and edit text documents.



```
   F:TXT2HTML.TXT      L00001  C01 Insert
                  N O N - D O C U M E N T   E D I T   M E N U
     CURSOR      SCROLL        ERASE      OTHER               MENUS
  ^E up       ^W up         ^G char    ^J help             ^K block & save
  ^X down     ^Z down       ^T word    ^I tab              ^P print controls
  ^S left     ^R up screen  ^Y line    ^V turn insert off  ^Q quick functions
  ^D right    ^C down      Del char    ^O set tab width   Esc shorthand
  ^A word left   screen     ^U unerase ^N split the line
  ^F word right             ^B top bit ^L Find/replace again
TXT2HTML  1.02  04/10/97
------------------------

TXT2HTML is an MSDOS program that converts ASCII text files to HTML files.
Files to be converted are typically documentation files with paragraphs
separated by blank lines (like this file).

TXT2HTML's conversion is modeled on the characteristics of typical
documentation files.  It is designed to convert paragraphs of text.
If your text file contains program source code or tabular data, you will
need to do some additional manual editing of the HTML file.  Tabular
data's formatting can be maintained by either delimiting the data with
<PRE> </PRE>, or by using the HTML 3 table features.

TXT2HTML will convert the special characters ", &, <, and > into valid
```

Find WordStar history here: http://www.wordstar.org/wordstar/history/history.htm

wordstar
quick reference card


ProgramKey
WORDSTAR®
MAIN MENU
OPENING/NO FILE MENU
COMMAND MENUS
^J Help Menu
ONSCREEN
PRINTER
QUICK
FILE
^O Screen Menu
^P Print Menu
^Q Quick Menu
^K Block Menu
DOT COMMANDS
MAIL MERGE

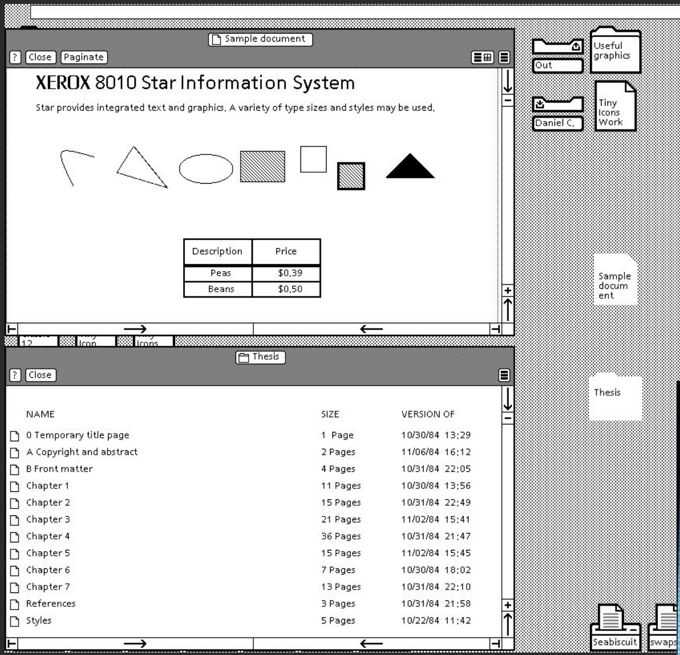
A few WordStar commands
(^ indicates one should hold down the Ctrl key)

Cursor left character	^S
Cursor right character	^D
Cursor down line	^X
Cursor up line	^E
Cursor left word	^A
Cursor right word	^F
Cursor beginning of line	^QS
Cursor end of line	^QD
Cursor top of screen	^QE
Cursor bottom of screen	^QX
Scroll down one line	^W
Scroll up one line	^Z
Scroll down one screen	^C
Scroll up on screen	^R
Delete character left	Del
Delete character right	^G
Delete word right	^T
Delete line left	^QDel
Delete line right	^QY
Delete entire line	^Y
Tab	^I
Insert CR	^N
Insert CR or go to next line	CR
Block beginning	^KB
Block end	^KK
Block copy	^KC
Block move	^KV
Block delete	^KY

Interested? Purchase a WordStar command emulator package for Microsoft Word by visiting www.wordstar.org

the future could usually be seen before it arrived

Xerox STAR, 1981



Sample document

? Close Paginate

XEROX 8010 Star Information System

Star provides integrated text and graphics. A variety of type sizes and styles may be used.

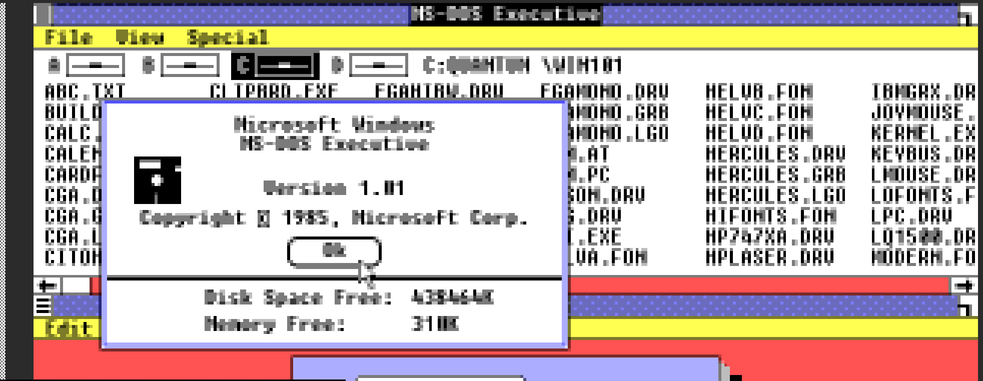
Description	Price
Peas	$0.39
Beans	$0.50

Thesis

? Close

NAME	SIZE	VERSION OF
0 Temporary title page	1 Page	10/30/84 13:29
A Copyright and abstract	2 Pages	11/06/84 16:12
B Front matter	4 Pages	10/31/84 22:05
Chapter 1	11 Pages	10/30/84 13:56
Chapter 2	15 Pages	10/31/84 22:49
Chapter 3	21 Pages	11/02/84 15:41
Chapter 4	36 Pages	10/31/84 21:47
Chapter 5	15 Pages	11/02/84 15:45
Chapter 6	7 Pages	10/30/84 18:02
Chapter 7	13 Pages	10/31/84 22:10
References	3 Pages	10/31/84 21:58
Styles	5 Pages	10/22/84 11:42

Microsoft Windows 1.01, 1985



```
C:QUANTUM \WIN101
ABC.TXT        CLIPBRD.EXE    EGAHIRW.DRV    EGAMONO.DRV    HELVB.FON      IBMGRX.DR
BUILD                                        MONO.GRB       HELVC.FON      JOYMOUSE.
CALC.                                        MONO.LGO       HELVD.FON      KERNEL.EX
CALEN                                        .AT            HERCULES.DRV   KEYBUS.DR
CARDF                                        .PC            HERCULES.GRB   LMOUSE.DR
CGA.D                                        SON.DRV        HERCULES.LGO   LOFONTS.F
CGA.G                                        S.DRV          HIFONTS.FON    LPC.DRV
CGA.L                                        .EXE           HP747XA.DRV    LQ1500.DR
CITOH                                        LVA.FON        HPLASER.DRV    MODERN.FO
```

Microsoft Windows
MS-DOS Executive

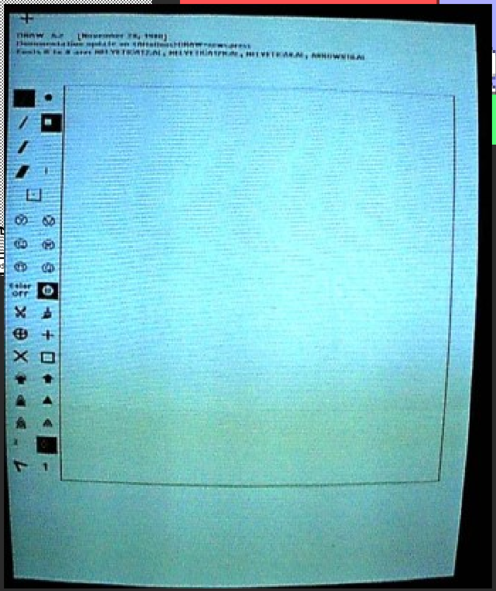
Version 1.01

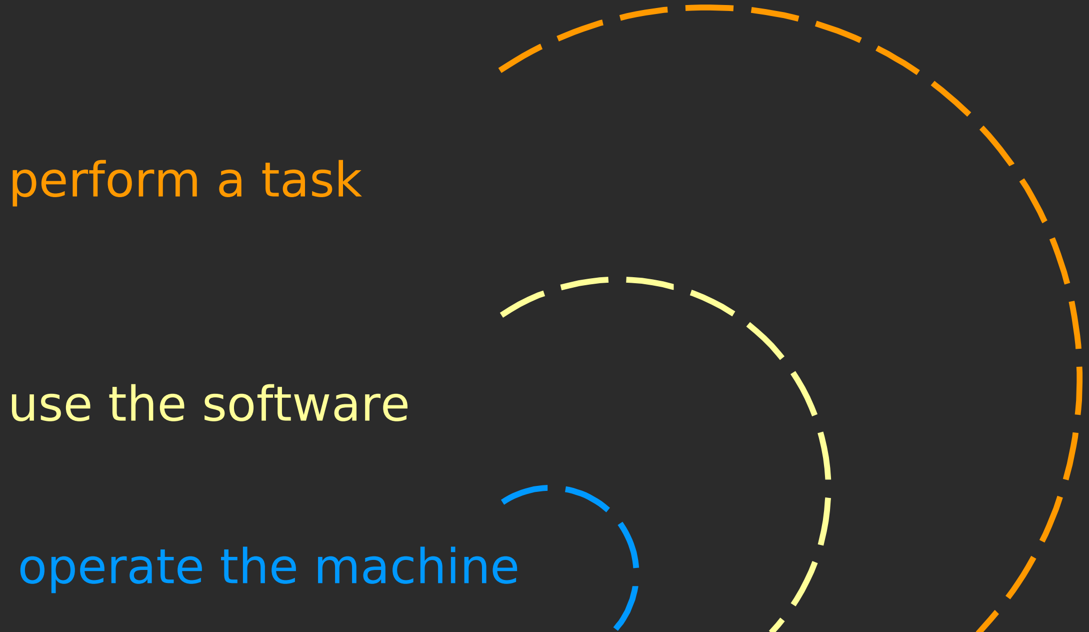
Copyright © 1985, Microsoft Corp.

Ok

Disk Space Free: 43846K
Memory Free: 31?K

Xerox Alto, 1972



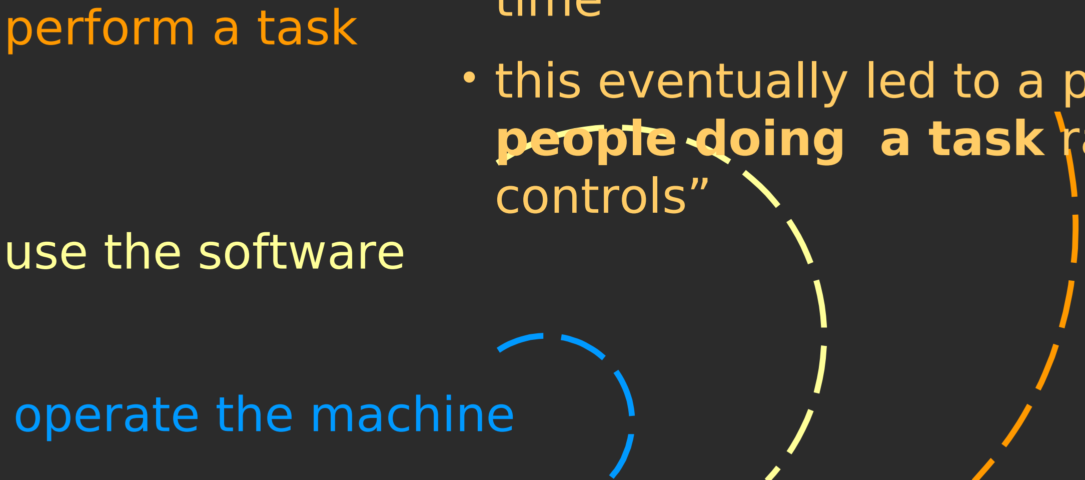


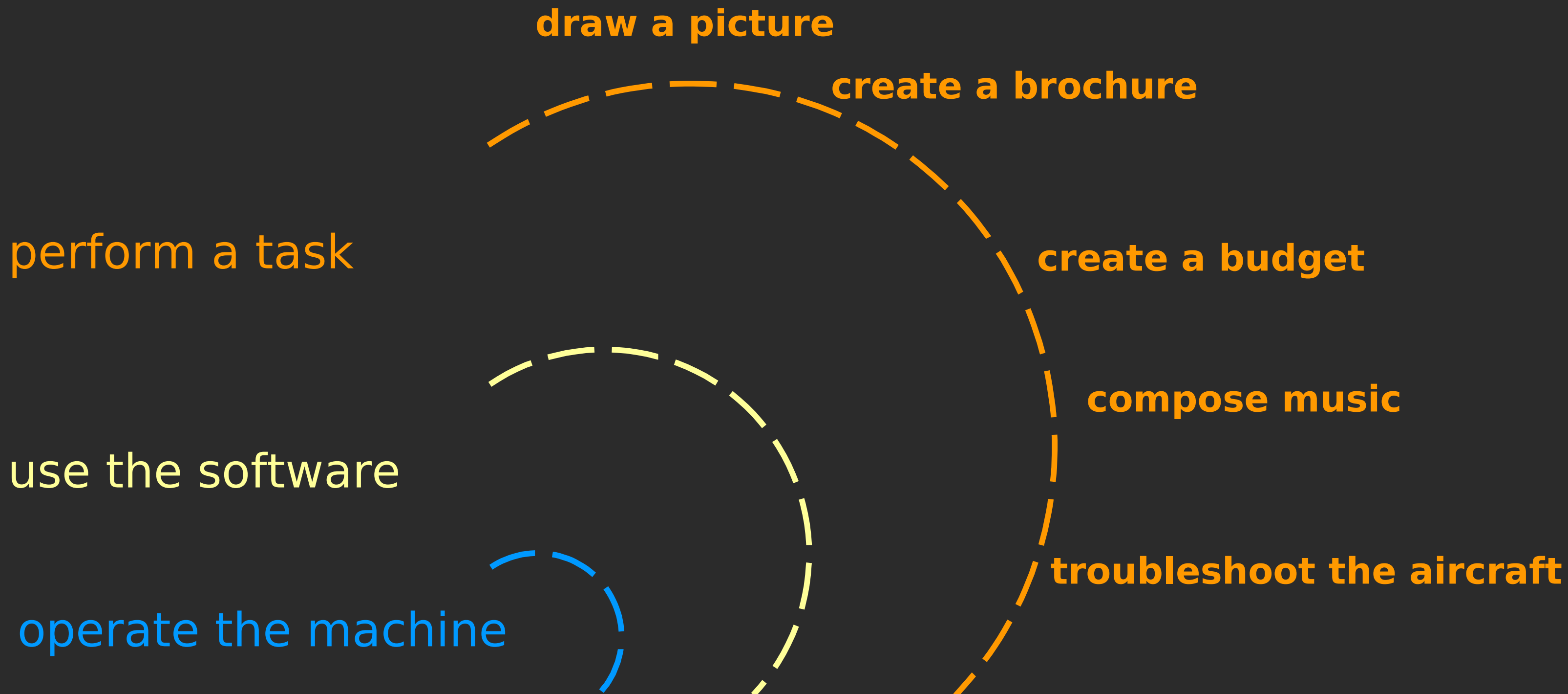
perform a task

use the software

operate the machine

- wordstar was so complex yet so popular, it invited both complaint and competition
- the success of Lotus 1-2-3 over Visicalc was partly due to ease of use and appropriate power (tip o’ the hat to Mitch Kapor); that and its enterprise-penetrating platform, the IBM PC
- its use in large companies led to an emphasis on ease of learning, ease of use, reduced errors, saved time
- this eventually led to a professional emphasis on **people doing a task** rather than “a tool with good controls”





draw a picture

create a brochure

perform a task

create a budget

compose music

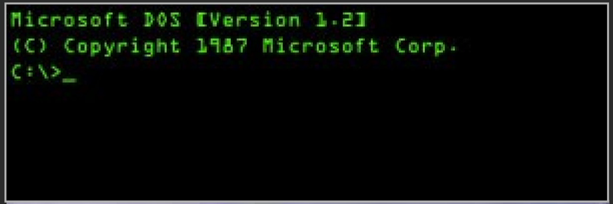
use the software

troubleshoot the aircraft

operate the machine

the mac taps into pent-up desire for ease and pleasure of use

Think of a world full of command-line interfaces…



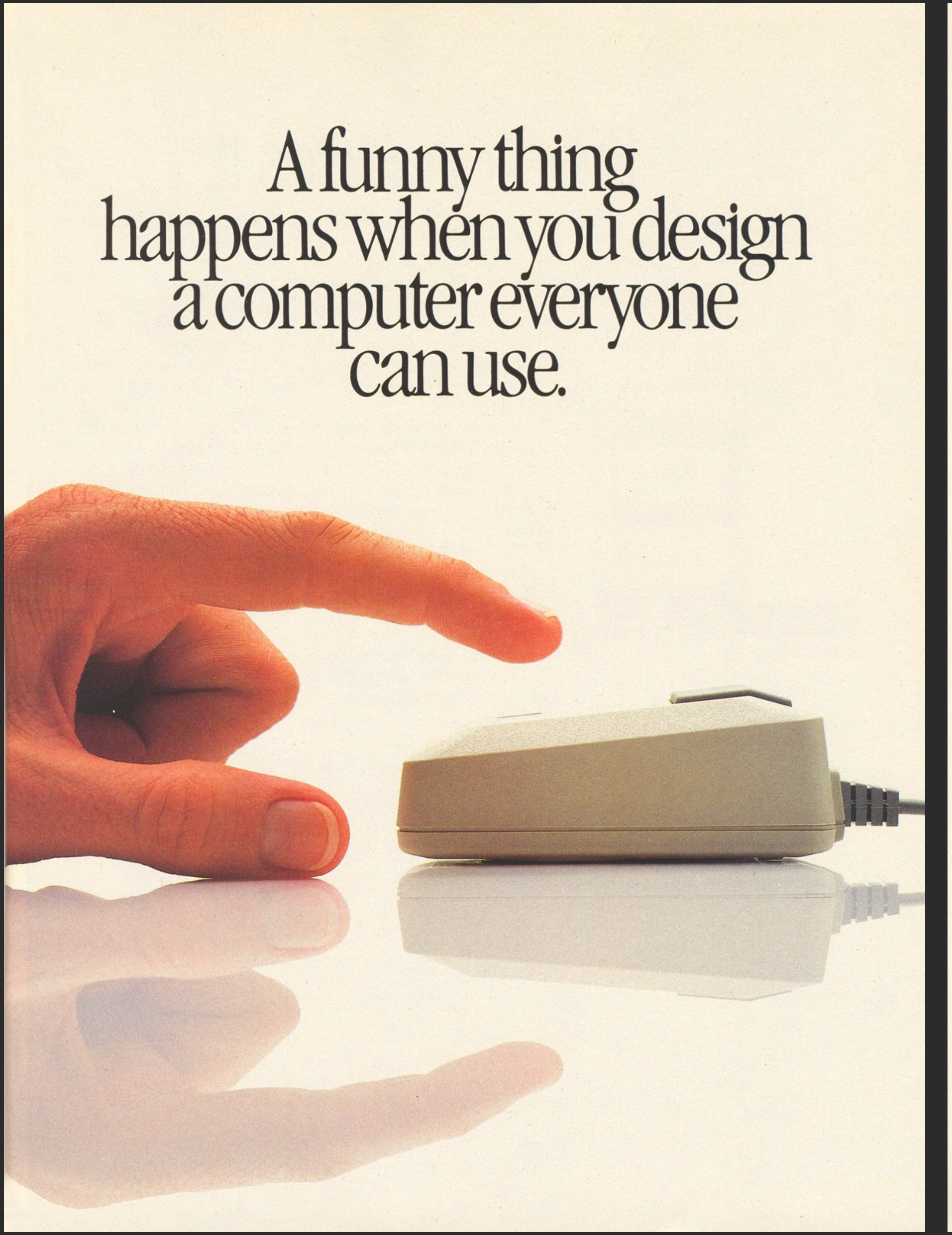
```
Microsoft DOS [Version 1.2]
(C) Copyright 1987 Microsoft Corp.
C:\>_
```

Then, during the 1984 Superbowl, you see the first commercial for the Macintosh (directed by Ridley Scott of Blade Runner fame). A crowd of solemn men is gathered in a gloomy auditorium, listening to a ranting bureaucrat on a huge screen.

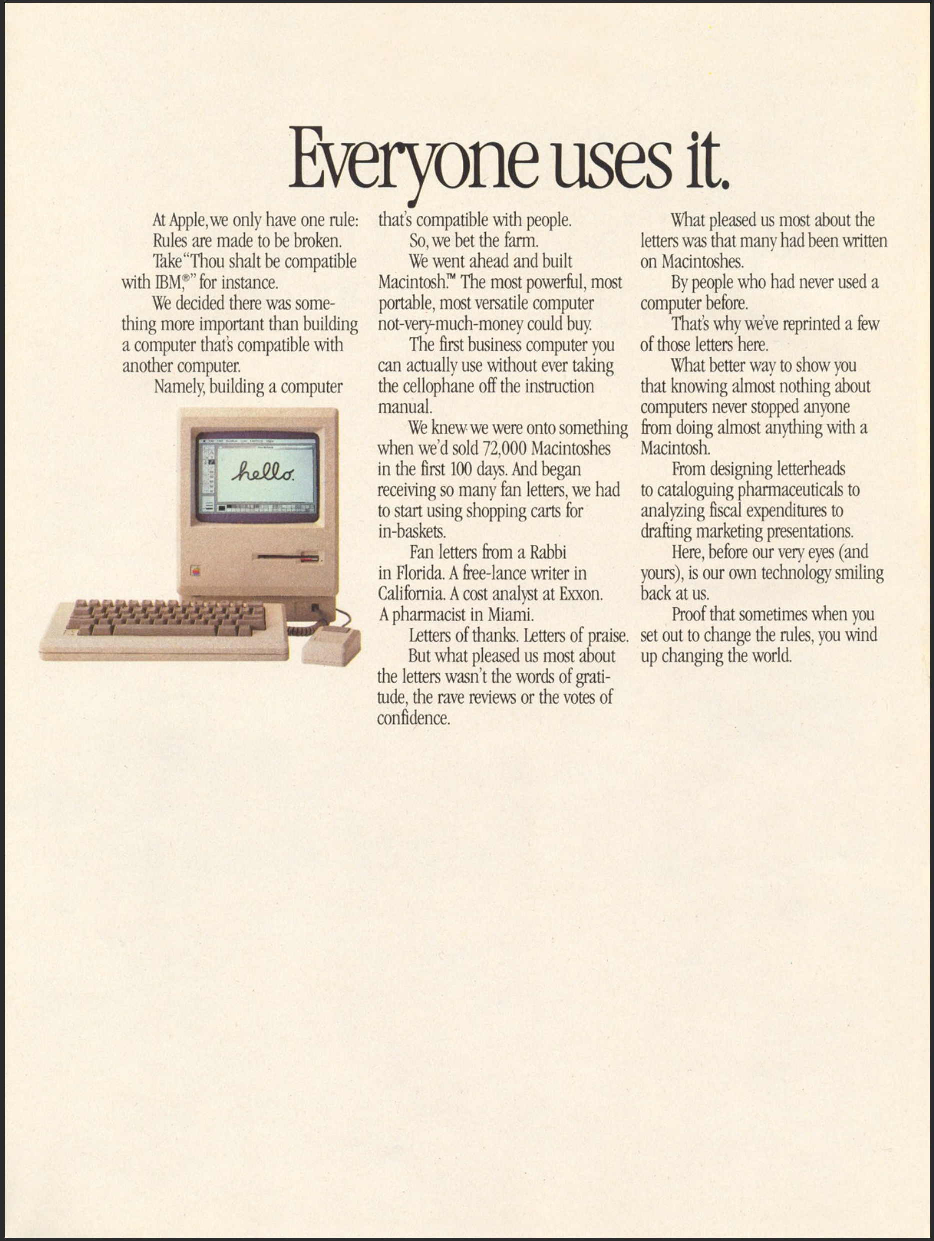
An athletic woman in colorful clothing runs into the auditorium, carrying a huge hammer…

…which she throws into the screen, smashing the image and voice of the status quo.

hello.



A funny thing happens when you design a computer everyone can use.



Everyone uses it.

At Apple, we only have one rule: Rules are made to be broken.

Take "Thou shalt be compatible with IBM,®" for instance.

We decided there was something more important than building a computer that's compatible with another computer.

Namely, building a computer that's compatible with people.

So, we bet the farm.

We went ahead and built Macintosh.™ The most powerful, most portable, most versatile computer not-very-much-money could buy.

The first business computer you can actually use without ever taking the cellophane off the instruction manual.

We knew we were onto something when we'd sold 72,000 Macintoshes in the first 100 days. And began receiving so many fan letters, we had to start using shopping carts for in-baskets.

Fan letters from a Rabbi in Florida. A free-lance writer in California. A cost analyst at Exxon. A pharmacist in Miami.

Letters of thanks. Letters of praise.

But what pleased us most about the letters wasn't the words of gratitude, the rave reviews or the votes of confidence.

What pleased us most about the letters was that many had been written on Macintoshes.

By people who had never used a computer before.

That's why we've reprinted a few of those letters here.

What better way to show you that knowing almost nothing about computers never stopped anyone from doing almost anything with a Macintosh.

From designing letterheads to cataloguing pharmaceuticals to analyzing fiscal expenditures to drafting marketing presentations.

Here, before our very eyes (and yours), is our own technology smiling back at us.

Proof that sometimes when you set out to change the rules, you wind up changing the world.

All 39 pages of advertising that Apple bought in a 1984 issue of newsweek are available here:
http://www.aci.com.pl/mwichary/computerhistory/ads/macnewsweek

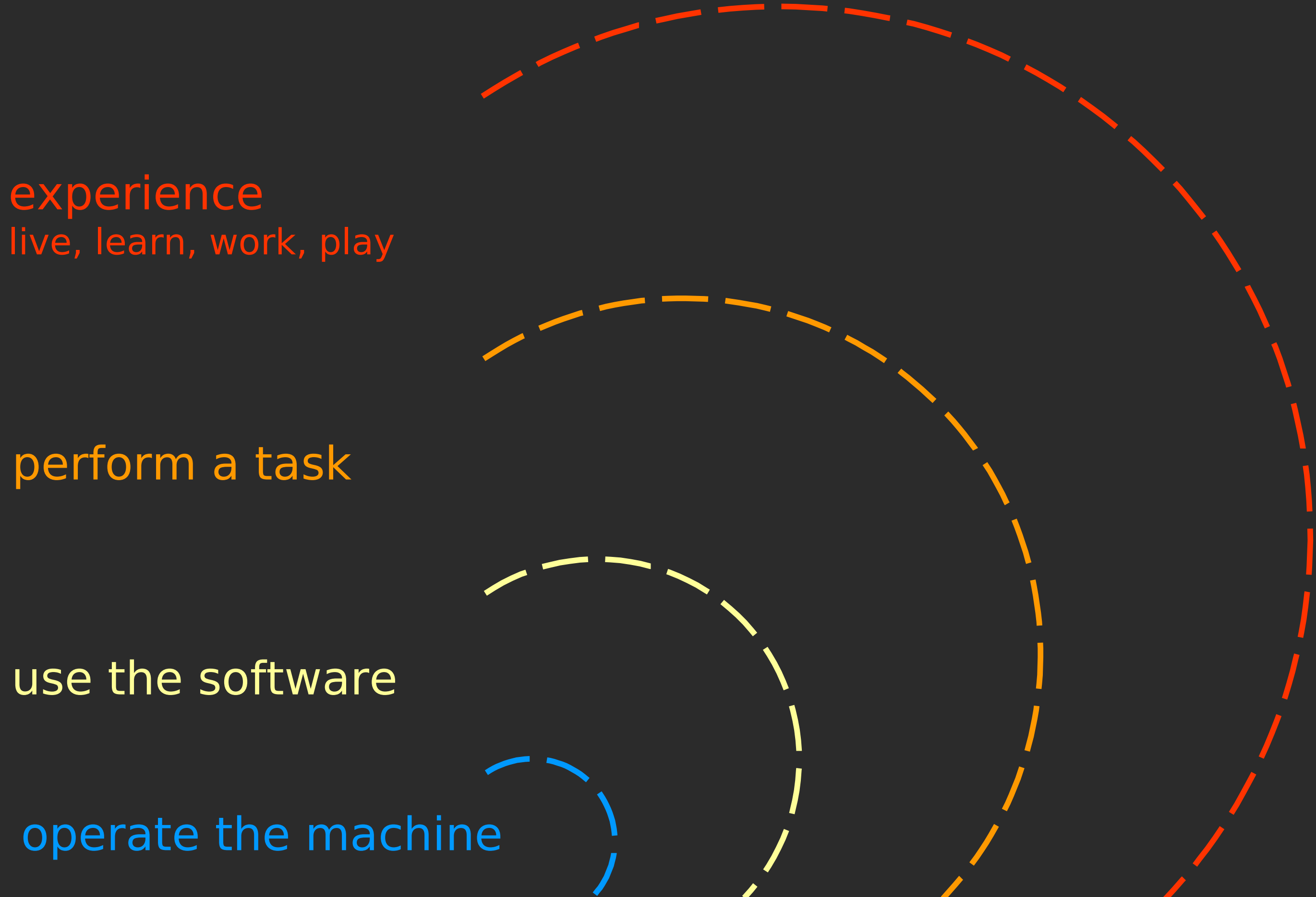
the software design manifesto

Mitch Kapor
1990

I don’t know that this was a landmark event in the whole industry’s eyes, but Mitch Kapor’s Software Design Manifesto was a clear articulation of the idea that making useful, usable, delightful software is a *design problem*, not an engineering problem. This is only a small extract here. The whole thing has both historical importance and modern currency.

“The Roman architecture critic Vetrivius advanced the notion that well-designed buildings were those which exhibited firmness, commodity and delight. The same might be said of good software. **Firmness**: a program should not have any bugs which inhibit its function. **Commodity**: a program should be suitable for the purposes for which it was intended. **Delight**: the experience of using the program should be a pleasurable one. Here we have the beginnings of a theory of design for software.”

www.kapor.com/homepages/mkapor/Software_Design_Manifesto.html present



experience

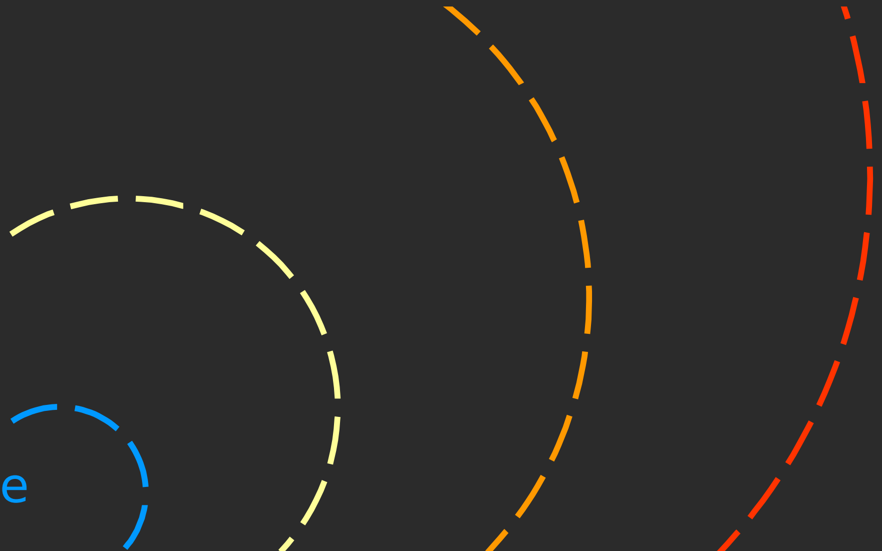
live, learn, work, play

perform a task

use the software

operate the machine

- after twenty years of trying to help people perform tasks, we realized success depended on expanding the scope of view
- most good work now involves an effort to fit context of use, characteristics of individuals, patterns of life
- most good work now attempts to go beyond expressed need to latent or masked needs



experience
live, learn, work, play

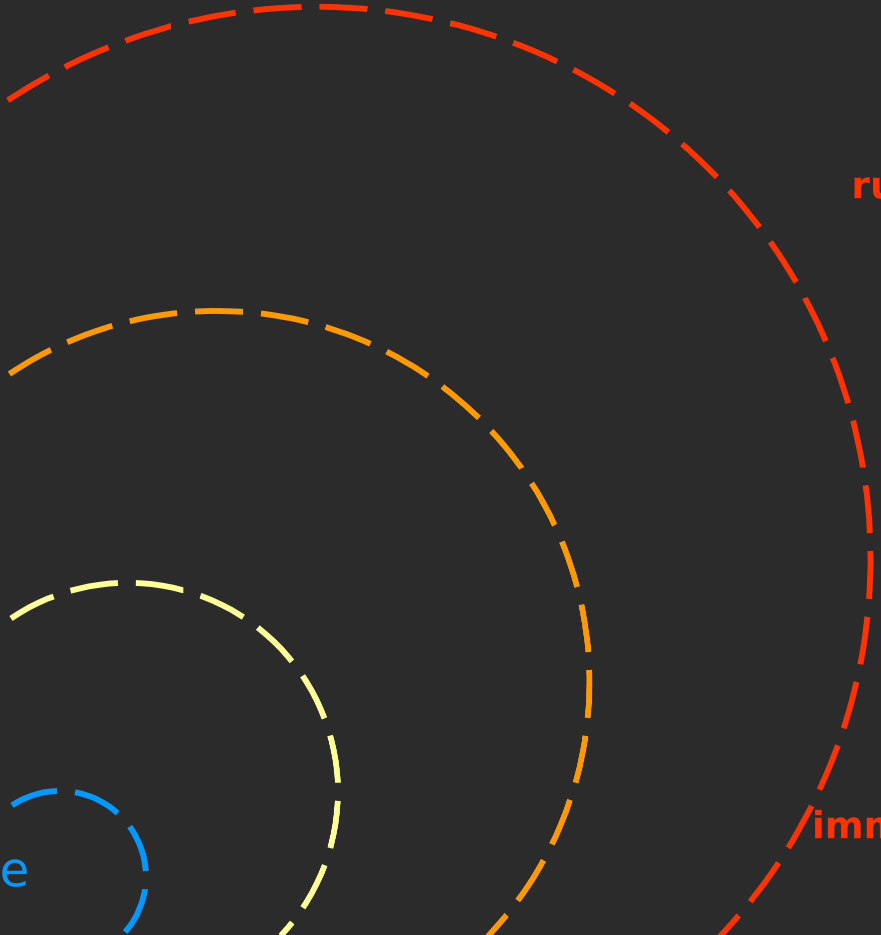
perform a task

use the software

operate the machine

manage a
household

compose music



experience
live, learn, work, play

perform a task

use the software

operate the machine

run a business

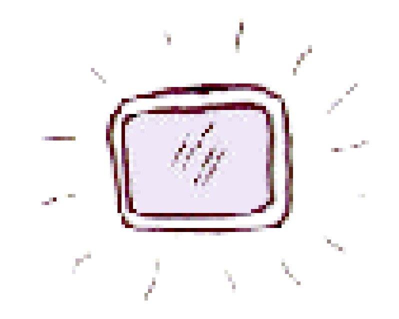
learn math

buy, use, &
maintain a car

immerse in a fantasy

art and engineering

To oversimplify, at the risk of stereotyping: if your primary concern is to make something cool or interesting happen on the screen, you are probably in the camp of artists or engineers. As opposed to...

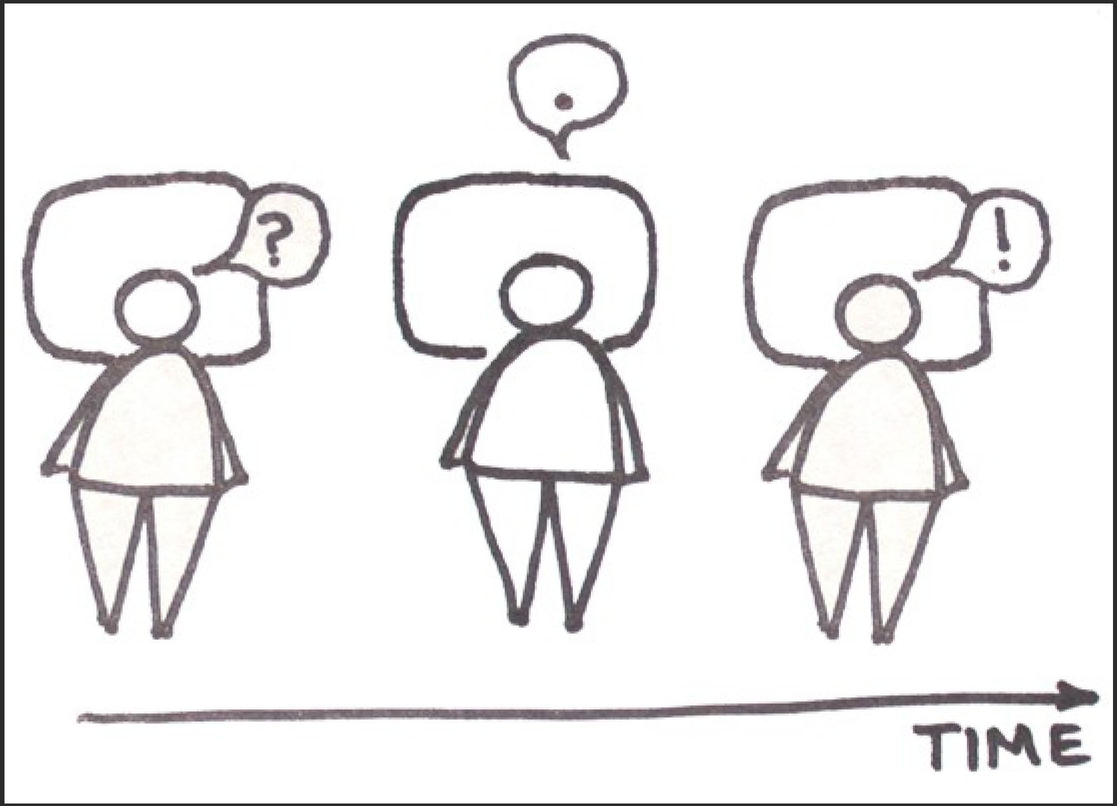


interface

...interface design, which is concerned with the person in front of the screen, with understanding and communication. But interface design often takes a fairly static view of things...

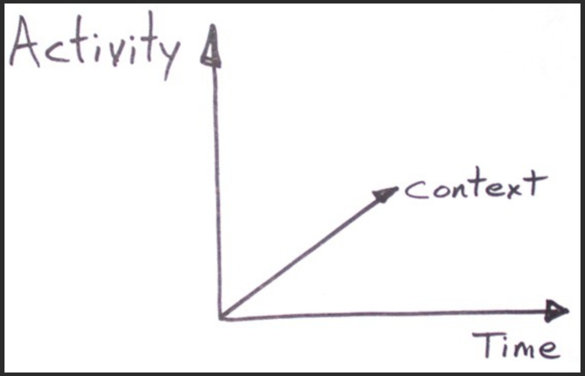
interaction

When we add time, we see the conversation back and forth between people and machines. We design the language for these conversations, we contribute something to the context in which they happen.



design to support a person doing an activity in context

To do a good job of interaction design, we have to understand as much as we can about the context, the activity, what else is going on, where people's attention is focused, what happens before and after, what their goals are, and so on.



design a vase

Shelley Evenson by way of Chris Pacione contributes this exercise to help us understand how interaction design these days differs from the days of “making tools.”

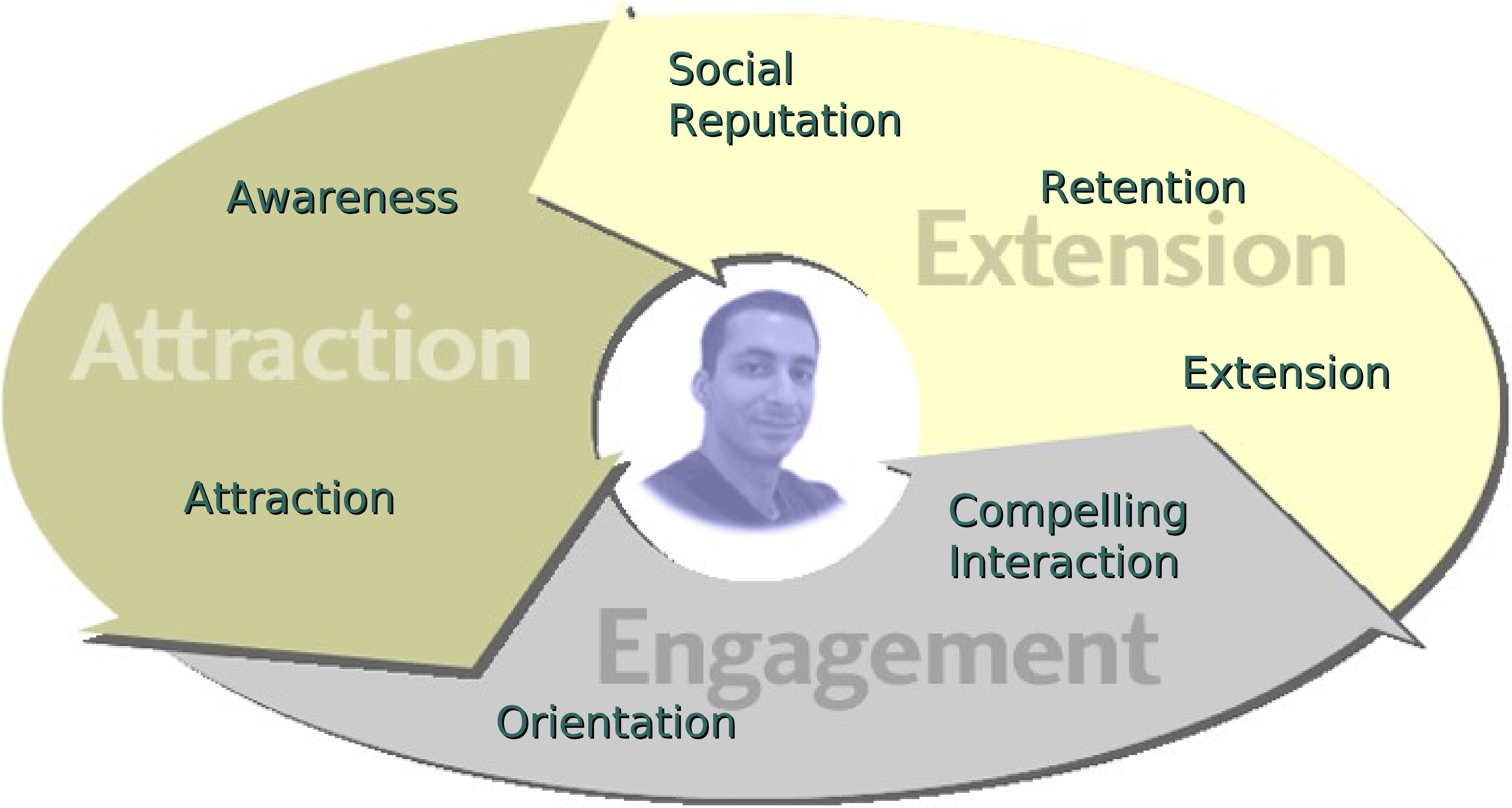
Suppose I asked you to design a vase. You would sketch or model any number of forms, most of them probably looking like a cousin of the vase shown here.

design a way to enjoy flowers

But suppose I asked you
to design a way for people to
incorporate plants into their
life, or a way for people to
enjoy flowers.

Contemporary design has
changed the questions.

the cycle of experience

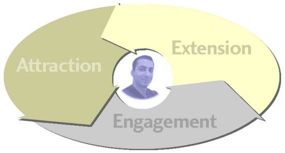


tip of the hat to john rheinfrank and shelley evenson

interaction design’s many layers of concern

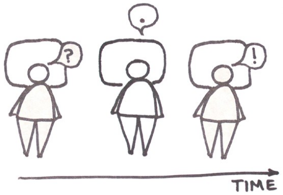
strategy

does the product connect with business goals?



experience

repeated interaction, activities in context

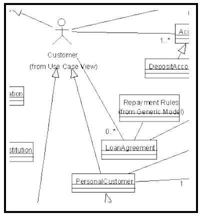


interaction

interface in use through time by different people

interface

presentation of information and controls



information & functionality

categories, types, attributes, relationships

the work: alignment

alignment

strategy
does the product connect with business goals?

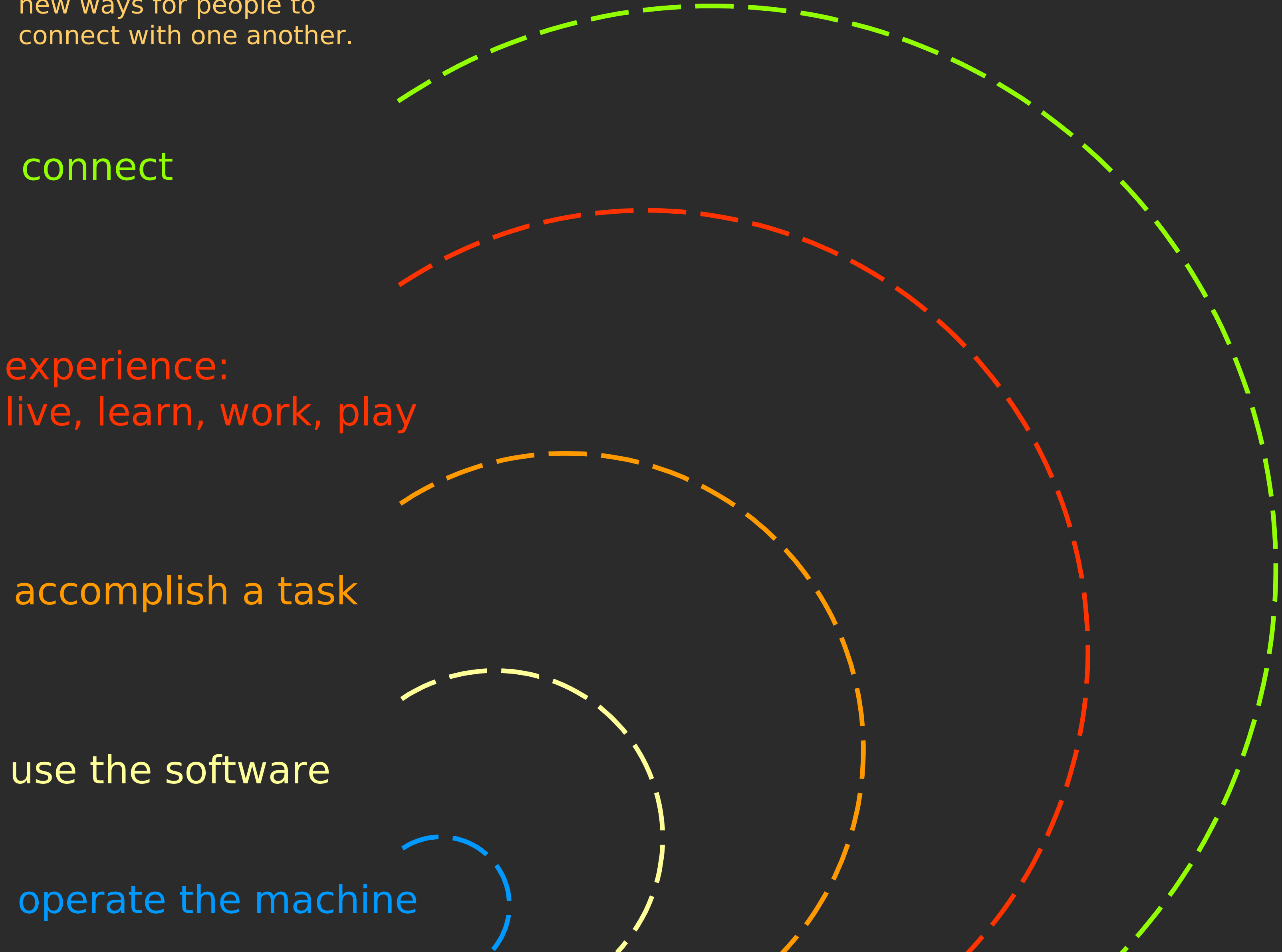
experience
repeated interaction, activities in context

interaction
interface in use through time by different peopl

interface
presentation of information and controls

information & functionality
categories, types, attributes, relationships

Another aspect of modern design: not only are we having to move past tools and objects to experiences, we are also learning to design new ways for people to connect with one another.



connect

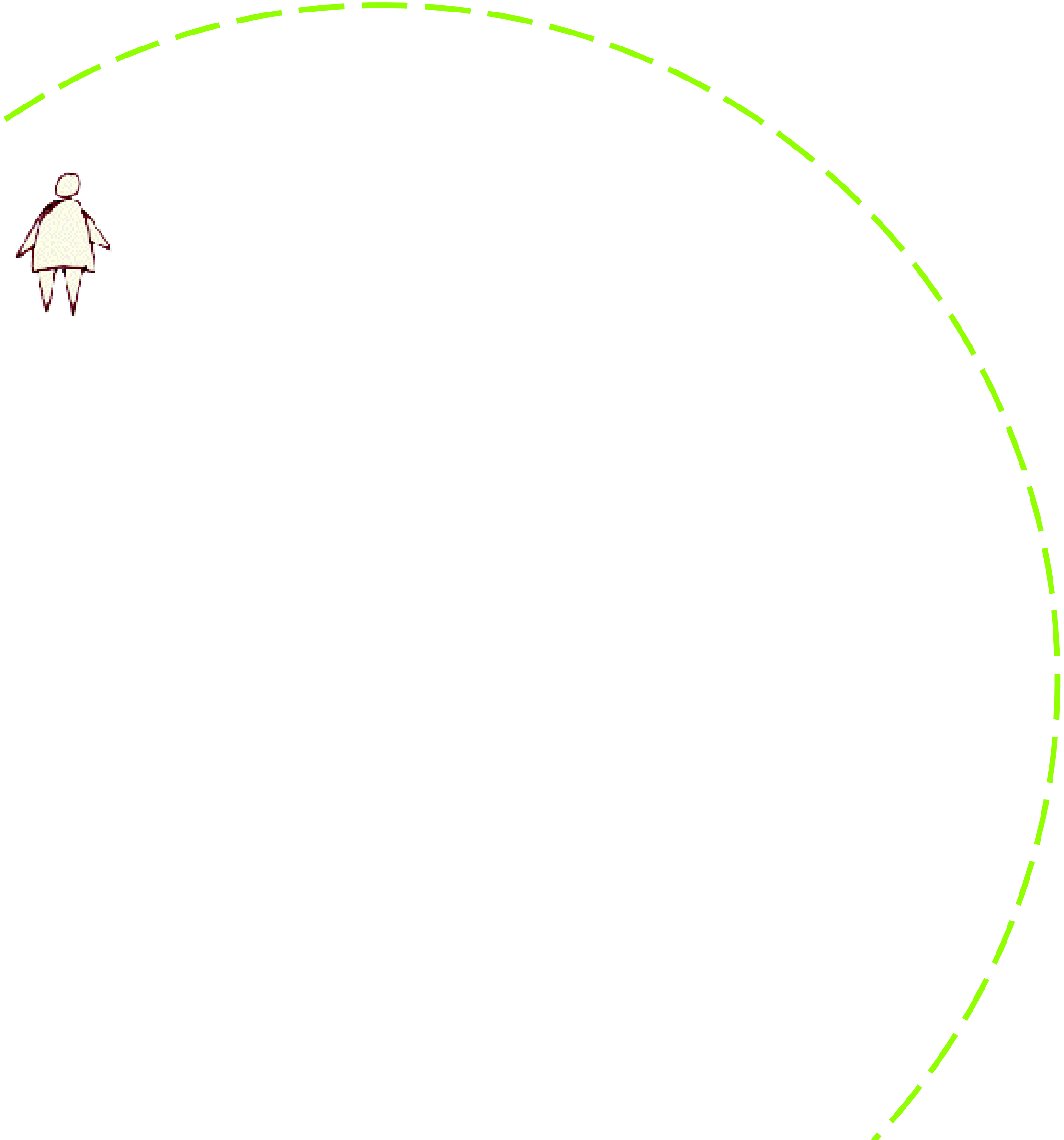
experience:
live, learn, work, play

accomplish a task

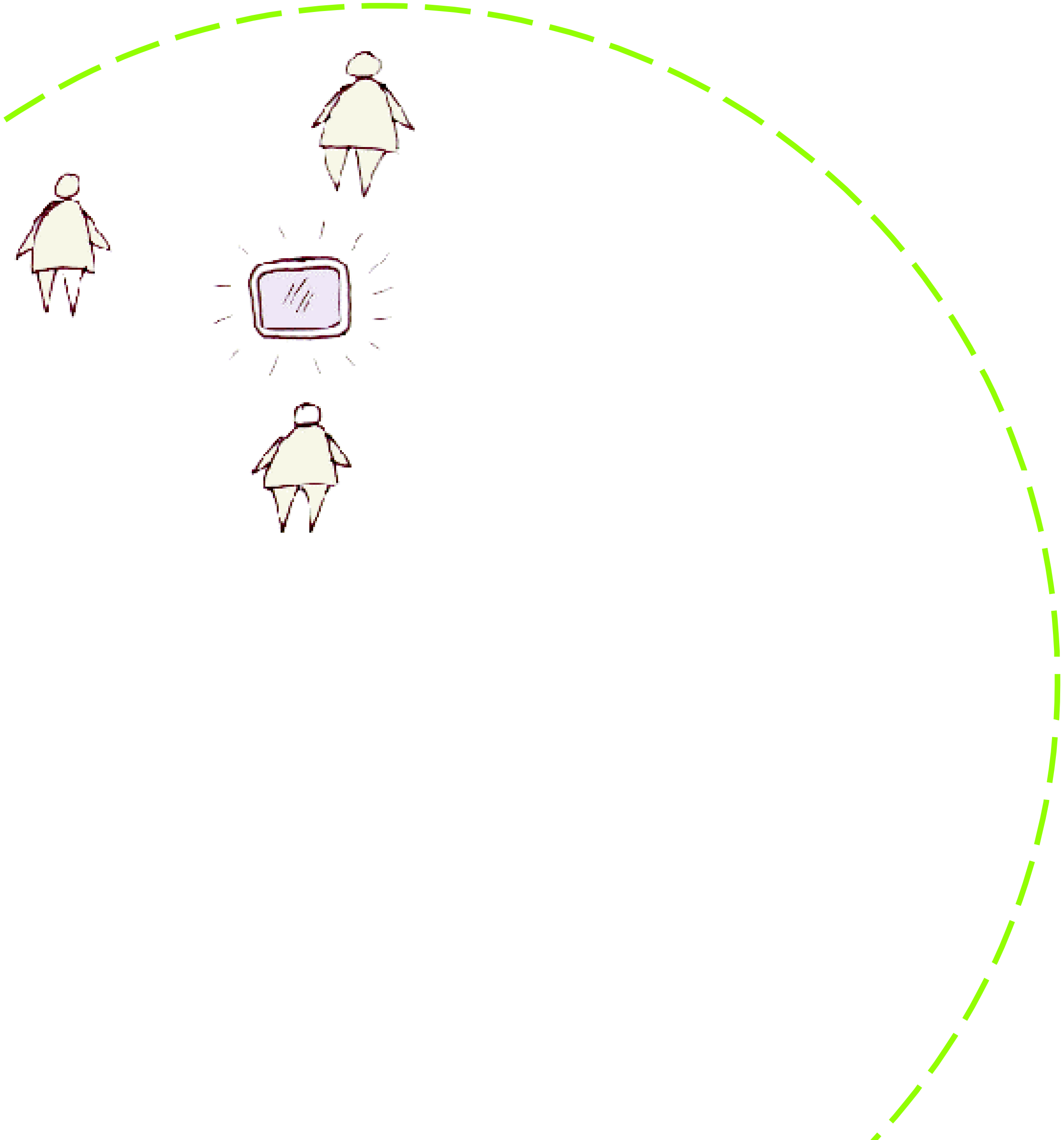
use the software

operate the machine

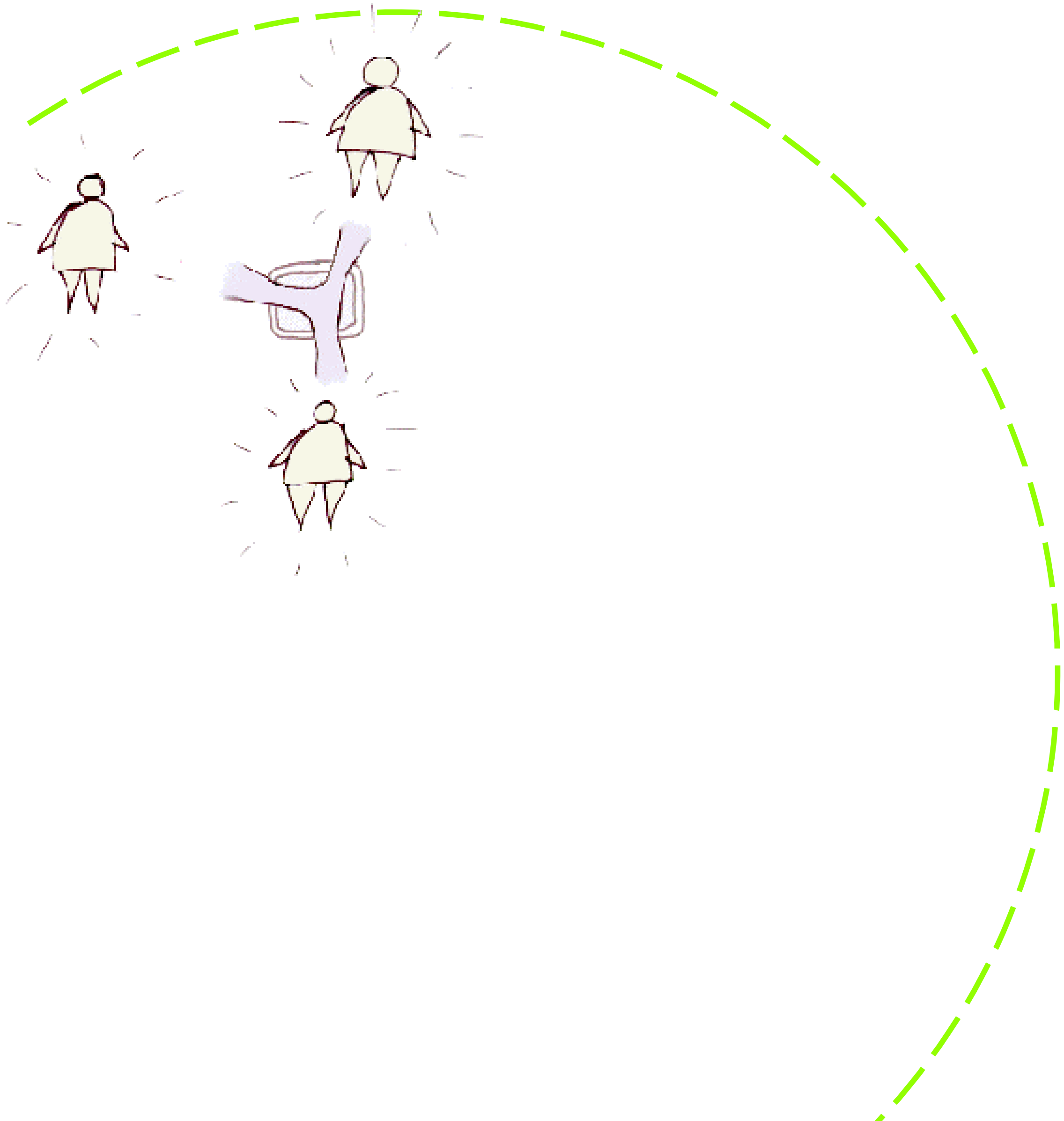
we are used to
designing for individuals



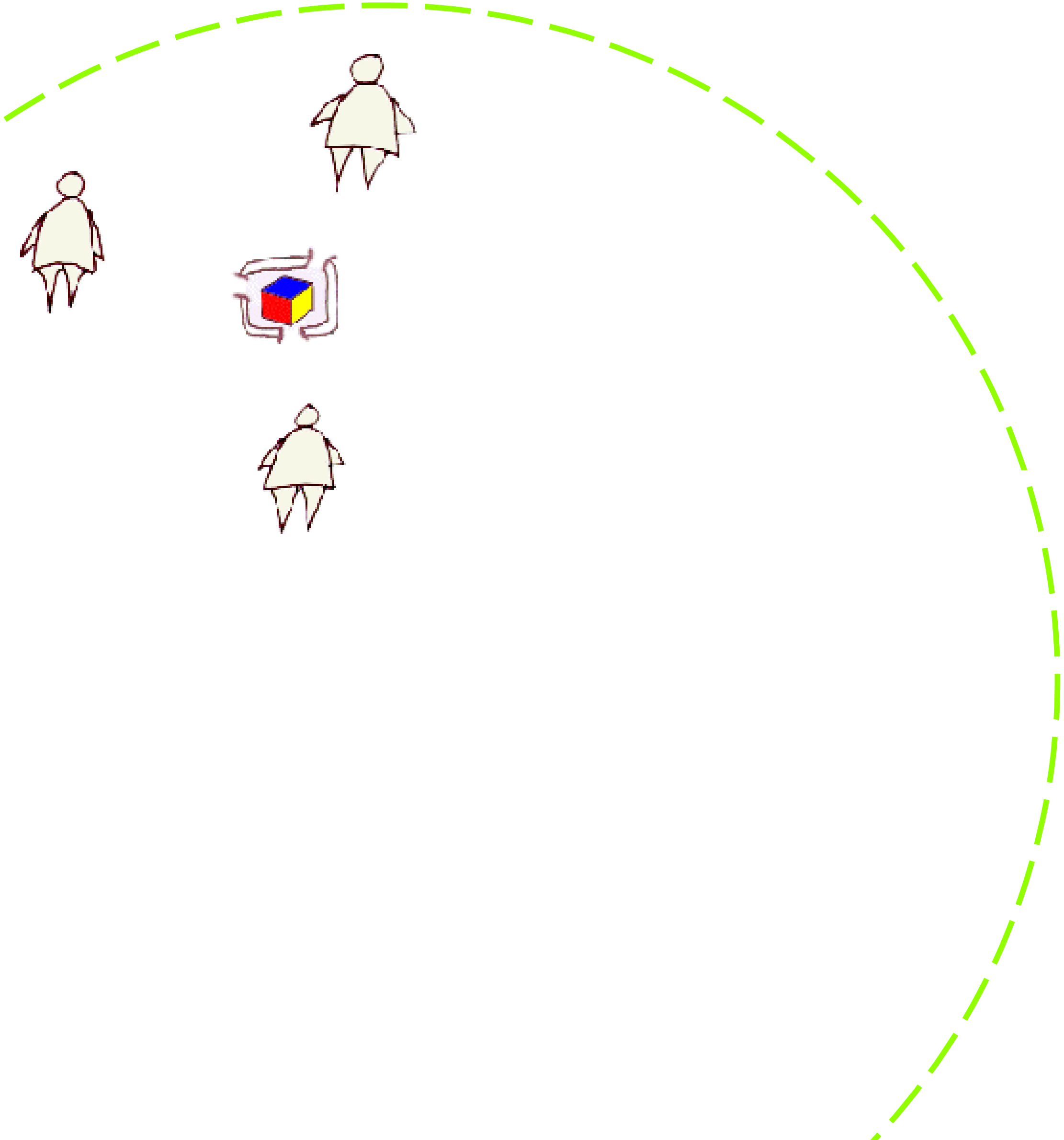
we’re learning to
account for differences
in individuals.
But the machines often
stay most prominent



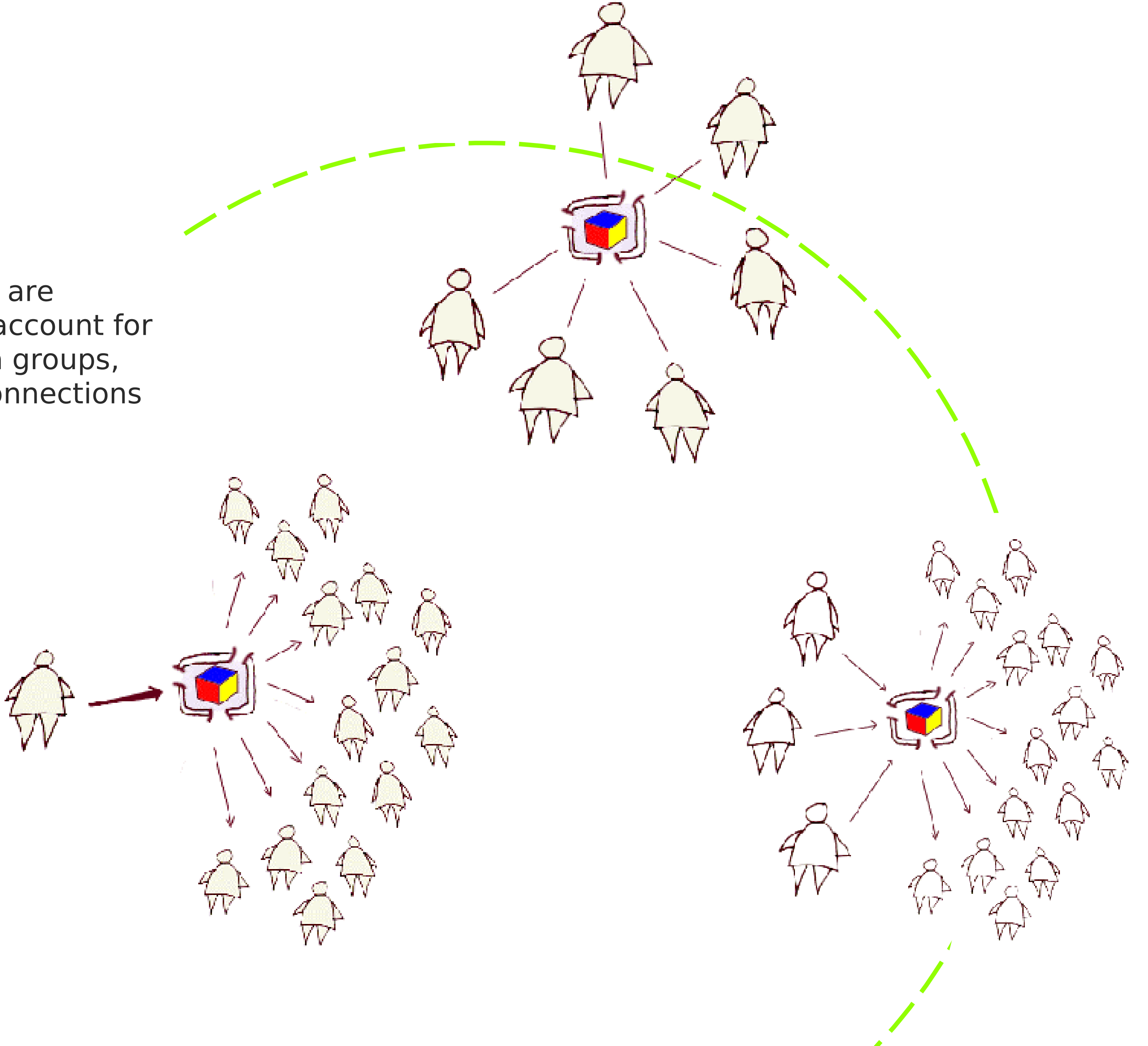
now, more often, the machines fade to the background –
we design for what happens between people *through* the machines



or we make it possible for people to build things together through the machines – the construction is in the foreground, not the technology



and now we are
learning to account for
variations in groups,
genres of connections



future

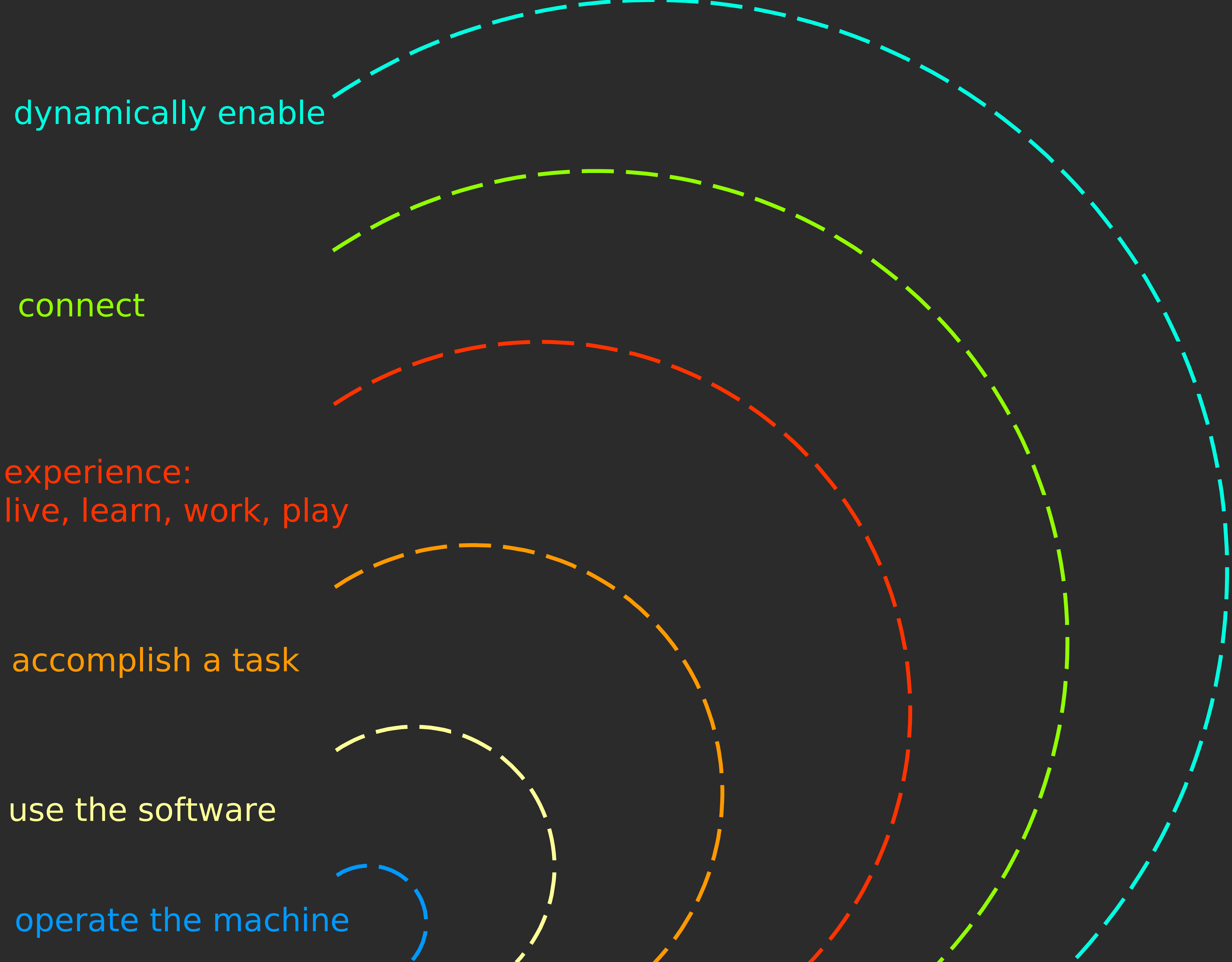
> “The future is already here.
> It’s just not evenly distributed.”
>
> — William Gibson

“The future is already here.
It’s just not evenly distributed.”

— William Gibson

“The past is still here.
It’s just not evenly distributed.”

— Me



dynamically enable

connect

experience:
live, learn, work, play

accomplish a task

use the software

operate the machine

dynamically enable

some are beginning to imagine (and work toward) a day when we complete the shift…

from “we adapt to the machines”

to “our environment and tools adapt to us as we move through life”

the best examples have yet to be imagined, but here are two I’ve heard mentioned, just so you get the idea:

- why do I have to convert this to pdf format to share it with you? why doesn’t the act of sharing imply some sort of conversion from authoring format to reading format?
- I like to listen to “Morning Edition” on the radio in the mornings. Why shouldn’t the broadcast follow me from bathroom to bedroom to kitchen, then into the car?

an explosion

- “the street has its own uses for things” (William Gibson)
- people are creating new kinds of interactions by themselves,
  bypassing the commercial product pipeline (e.g., web logs and peer-to-peer content sharing networks)
- people making a living, even getting benefits on EBay
- 79th world economy in 2003 = Ultima Online
  www.wired.com/wired/archive/11.01/gaming.html

one implication: we should learn how to give people “clay” or “legos” instead of products and services.

another implication: platforms are *very* important, from all points of view – design, technology, business

the future

- ecosystems of devices
- invisible computing – the machine has faded into the woodwork: where’s the interface?
- dynamic, distributed content
- dynamic structure
- dynamic form
- dynamic, distributed context
- dynamic, distributed audience
- dynamic, distributed use

the way forward

these technologies could bring pleasure, health and growth to people. They could also bring confusion, division, annoyance, oppression, death.

- apply empathic design practices
- always an interdisciplinary, collaborative process
- prototype and fail early and often
- beware the inevitable surprise of negative side effects
- maintain a humble attitude of discovery, exploration, service
- in choosing projects and doing your work, aim to create things that people really value, that make a positive contribution to areas of life that really matter.

mrettig@well.com

thank you

some resources on the history of computing

Besides the links sprinkled through thes slides, these are some of the resources I found helpful in preparing this presentation. My thanks to the many creators.

- www.computinghistorymuseum.org – lots of nice links and presentations.
- www.computer.org/pubs/annals/annals.htm – site for the IEEE journal, Annals of the History of Computing
- Want to see what it was like to run the world's first stored-program computer? Try the simulator at www.dcs.warwick.ac.uk/~Eedsac/
- www.tcm.org -- the computer museum
- users.libero.it/fmaida/emusearch – a place to search for all sorts of emulators. Relive the past!
- A great collection of images: www.tonh.net/museum/index.html
- 1993 SIGCHI Conference video, which has the Sketchpad demo.
- LOOP interview: Archiving Experience Design (loop.aiga.org)