Fundamentals of Situated Interaction

Wendy Mackay & Michel Beaudouin-Lafon
14 January 2021

mackay@lri.fr          mbl@lri.fr
Human-Computer Partnerships

or

Co-Adaptive Instruments
Computer hardware has changed dramatically over the past 40 years ...
Key Challenge

How can we improve interactive systems, given today’s ever-increasingly complex computational environment?
We have multiple relationships with computers

Computer as a **tool**
I accomplish the task myself

Computer as a **servant**
It accomplishes the task for me

Computer as a **medium**
It lets me communicate with other people
Graphical User Interfaces

Designed for executive secretaries to process documents in a completely different technology environment.

Dates back to the 1970s to:
copy hand-written notes
check for mistakes
format on letterhead

Problem:
Brilliant then,
out-moded today
GUIs are a vindication ... and a challenge

Human-Computer Interaction research fought hard to make interfaces easier to use.

Today, novices easily accomplish simple tasks.
GUIs are a vindication ... and a challenge

Human-Computer Interaction research fought hard to make interfaces easier to use.

Today, novices easily accomplish simple tasks.

Yet ... advanced research in interaction techniques is rarely adopted in commercial systems.

Today, experts use inefficient techniques and are constantly forced to change their behavior.
Desksops, the web and apps ...  

Require constant relearning:
- each new version introduces arbitrary changes
- each system requires slightly different interaction

Require high visual attention
Do not scale
Depend on specific devices
We need to reassess human-computer interaction

Early assumptions about graphical user interfaces no longer hold

Everyone, not just experts
manages increasing quantities of data
faces information overload
constantly relearns the details of interaction

Redefine what we mean by “computer literacy”
Human-Computer Relationships

Between people and physical tools:
  follow well-known physical principles
  users can learn them
  users can appropriate them
Human-Computer Relationships

Between people and physical tools:
   follow well-known physical principles
   users can learn them
   users can appropriate them

Between people and computer tools:
   follow arbitrary constantly changing rules
   users must learn, and relearn, and relearn them
   users break them when they try to appropriate them
Learning to play a musical instrument—from novice to virtuoso—the instrument becomes part of the body.
Compare to learning software: every ‘upgrade’ changes the interface tools belong to the application, not the user.
Co-adaptive Instruments

Worthwhile spending time and energy learning them

Complex tools become accessible
  can learn cognitive and sensori-motor skills
  can adapt to new situations

Move beyond
  graphical user interfaces
  to expert instruments

To do this:
  Extract widgets from applications
  to create personal instruments
Human-Computer Partnerships
What do we mean by ‘partnership’?

Take a taxi
Driver in control
What do we mean by ‘partnership’?

Take a taxi
  Driver in control

Drive a motorcycle
  User in control
What do we mean by ‘partnership’?

- Take a taxi
  Driver in control

- Drive a motorcycle
  User in control

- Ride a horse
  Shared control
A ‘simple’ human-computer partnership

User types – Google suggests – User chooses
Focus on interaction, not interfaces

How can we let users control interaction in a flexible, reusable way?

How to develop expertise without constantly relearning skills?

Co-adaptive Instruments
Separate *interaction* from data and functionality
Interaction becomes a first-class object
Key phenomenon: *Co-adaptation*

Users *adapt* to a new system they *learn* to use it

Users *adapt* the new system to their own needs they *appropriate* and change it
Co-adaption

Inspired by co-evolution in biology
Organisms create their environment
even as they adapt to it

Anaerobic bacteria change the atmosphere
making it possible for aerobic bacteria to emerge

Users change spreadsheets from an addition tool
to a tool for exploring ‘what if’ scenarios
Reciprocal Co-adaptation

People adapt their behavior to technology
  ... they learn it
People adapt the technology for their own purposes
  ... they appropriate it

Computers adapt their behavior to people
  ... machine learning
Computers adapt human behavior
  ... training
Our vision:

Software tools should be incrementally learnable

People should choose and control their own tools

Software tools should be easy to appropriate
Dynamic partnership: Progressive algorithms reveal intermediate recognition states

Octopocus: Learning complex gestures
Octopocus: Learning complex gestures

Experts *just do it*
Octopocus: Learning complex gestures

Experts *just do it*
Novices *hesitate* … which activates:

- **feedforward** shows current available gestures
- **feedback** shows what the recognizer sees
OctoPocus is a dynamic guide providing continuous feedforward and feedback that helps users to execute gesture-based commands.
Physical tools are easy to appropriate — software tools are not
Arpege: Learning chords on a multi-touch surface

Beyond one- and two-finger gestures:

- novice to expert transition
- feedforward and feedback
Arpege: Design and learning of multifinger chord gestures
Dynachord: Combining chords and gestures

Chord sequences for a larger chord vocabulary
Dynamic adjustment of parameters
Dynachord

Enter a chord with one hand to choose a color

Continuously adjust the color with the other hand
How can we help users choose and control their own tools?
Appropriation

Interaction designers usually assume that users will focus on their system and use it as intended.

Users often use systems in different ways:
- They may have a different mental model of the system.
- They may turn ‘mistakes’ into opportunities.
- ‘Bugs’ become ‘features’.

Anything that involves communication among people is usually adapted for new purposes.
How can we help users appropriate technology?

Creating a partnership in which the user defines the **semantics** of the interaction with the computer

**Interaction Browser** : Linking marks to actions
**Knotty Gestures** : Interacting while writing
**Musink** : Creating a user-defined language
**Façades** : User-reconfigurable interfaces
Interaction browser: User-defined commands

Air traffic controllers annotate flight strips
Marks can be linked to RADAR and other computer functions
Users define what marks mean
Striptic

Flights in my Hands: Coherence Concerns in Designing a Tangible Space for Air Traffic Controllers, (Letondal et al., CHI'14)
Knotty Gestures

- Draw a dot, define a command
- Interact while writing
- Interact with command later
Knotty Gestures

Interactive Paper
Users interact as they write
or define their own gestures
and interact with them later
Knotty Gestures: Creating an interactive controller

Draw a line with a ‘knotty gesture’ at the end

Choose “recording” to define the type of line
Define where the recording will start
Define an end point for the recording
Knotty Gestures: Creating an interactive controller

Slide the pen along the line to move forward or backward on the recording
Drawing a Math Calculator

This line acts as a base for attaching mathematical value sliders

The knotty gesture at the end defines the type
Any knot drawn on line lets the user select a mathematical function
The extensions act as value controllers. Sliding the pen over the line moves through the range of function values, shown on the pen display.
Knots may define ranges or act as traces of past interactions with specific values.
<table>
<thead>
<tr>
<th>Day</th>
<th>Lecturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>Dr. Smith</td>
</tr>
<tr>
<td>Tue</td>
<td>Peter</td>
</tr>
<tr>
<td>Wed</td>
<td>6am</td>
</tr>
<tr>
<td>Fri</td>
<td>11am</td>
</tr>
</tbody>
</table>
But recognition is not the only problem …

Recognition must be *good enough*
but users override and reinterpret
no single ‘correct’ interpretation
recognized and non-recognized gestures co-exist

Real question:
Can *Musink* support the creative process?
What are the design implications for *Musink v2*?
Semi-Structured Delayed Interpretation

Key insights:
Spatial structure on paper
- improves recognition
- under user’s control
Recognition need not be immediate
- users decide when to interpret
interpretation changes over time
Musink

Musicians create their own musical languages on paper

... and go back and forth between paper and computer
MusInk

Define meaning of gestures over time

User decides if and when to interpret each gesture
Create interactive annotations
Reclassify a ‘squiggle’ and turn it into a trill
From symbols to wave forms:
Interpret a tremolo gesture as a waveform by OpenMusic
Transform structures into software representations
Leonard draws a new type of crescendo (score printed on Anoto paper)
Musink: Semi-structured, delayed interpretation

Users decide when and how each annotation should be interpreted by the computer

- score pointers
- textual elements
- scoping gestures
- connectors
Façades: Reconfiguring interfaces

Users can adopt parts of any Linux interface and reconfigure it for specific needs.

Grab three selections from GIMP and choose a brush and create a new, custom-made palette.
Substrates

Define the structures and rules
Ways to interpret the data
Different structures
to facilitate interpretation
Paper Substrates: create own language & structure

Composers create new structures for interpreting and composing music
Composers create their own reusable structures
Paper Substrates

A substrate is both an instrument for interpreting a personalized language and an object in its own right.
Interactive Paper Substrates for music composers
Paper Tonnetz
Draw music based on musical relationships among pitches
PaperTonnetz
Supporting Music Composition with Interactive Paper

Jérémie Garcia, Louis Bigo, Antoine Spicher and Wendy E. Mackay
INRIA, IRCAM, LACL
Paper Substrates
Composer create their own reusable musical structures
establish relationships among them
Arrange and Link substrates
Arrange and Link substrates to composition software.
Interactive Paper Substrates
to Support Musical Creation

Jérémie Garcia, Theophanis Tsandilas, Carlos Agon & Wendy E. Mackay

INRIA, Université Paris-Sud, CNRS, IRCAM & Stanford University
Quid Sit Musicus
Philippe Leroux

13th century musical scores
Each note indicates expression
Interact with early manuscripts for both performance and composition

Composer: Philippe Leroux
Researcher: Jérémie Garcia
Quid Sit Musicus
(composer: Philippe Leroux)
How do we create human-computer partnerships with mobile devices?

- Expressive Keyboard
- Fieldward
- CommandBoard
People can
adapt to technology
adapt the technology
they learn it
they appropriate it
People have rich cognitive and sensory motor capabilities increasingly, so do computers.

Why is the interface so limited?
Smartphones are easy ... but not powerful
Smartphones are easy ... but not powerful

What about creativity and expression?
Expressive keyboard

Gesture typing uses gestures to input text but focuses on finding one correct word
Expressive Keyboard
CommandBoard

Type and execute

Gesture shortcuts

Octopocus
Fieldward
Unified principles of interaction

Two complementary perspectives:
System: How to build it?

*Instrumental Interaction and Substrates*
Unified principles of interaction

Two complementary perspectives:
System: How to build it?
   *Instrumental Interaction*
   *and Substrates*

Human: How to interact with it?
   *Co-adaptive Systems*
   *Human-computer partnerships*