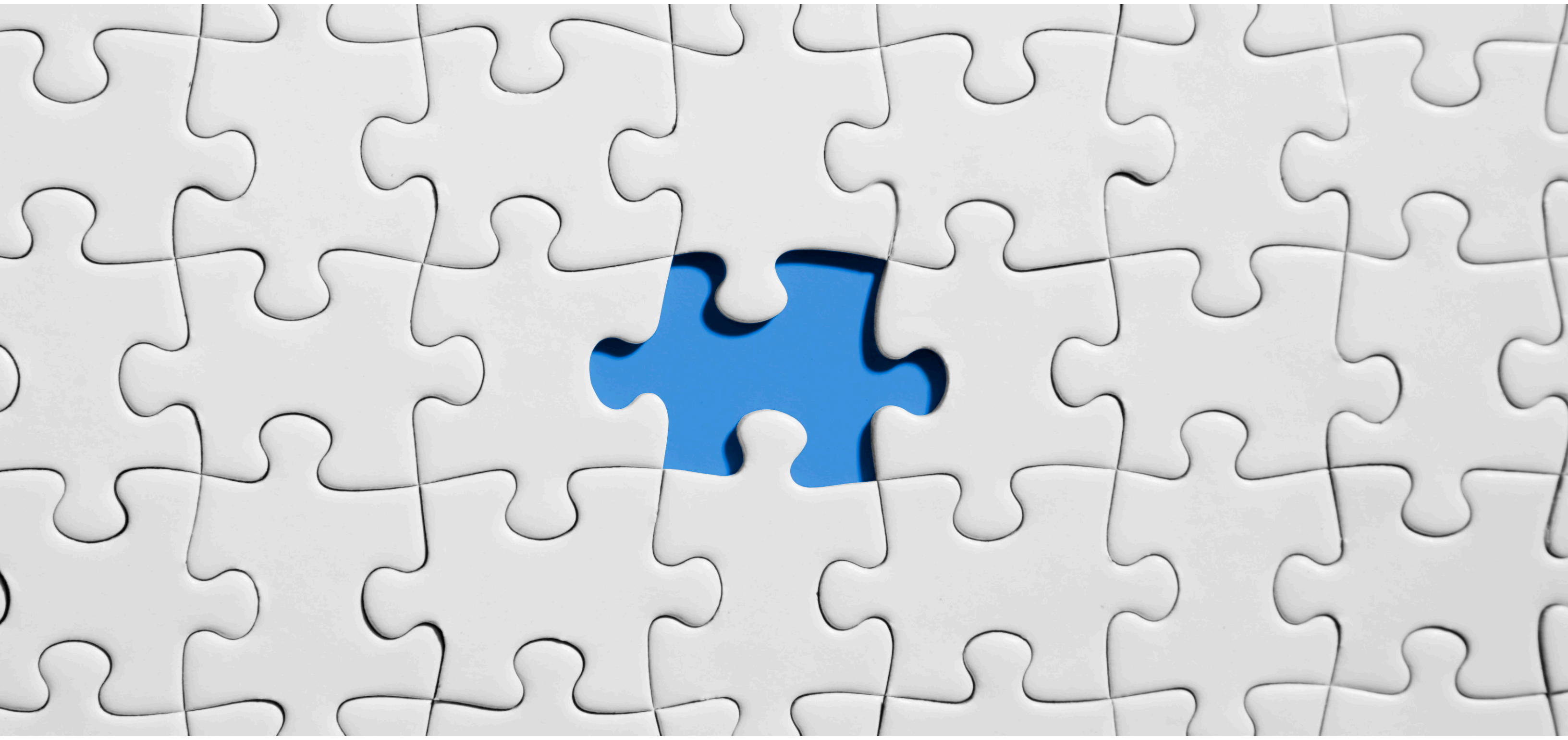
HUMAN-COMPUTER INTERACTION AND THE USER INTERFACE



Bruce Yellin
Data Center Architect
bruceyellin@yahoo.com

Knowledge Sharing Article
© 2018 Dell Inc. or its subsidiaries.

Table of Contents

Introduction 3

What is an Interface? 3

Interface Evolution 4

  Horse 4

  Car 4

  Elevator 5

The Evolution of Computer Interfaces 5

  The Punch Card 6

    Hanging chad 7

  The Keyboard 8

  The Mouse 8

  The Touchscreen 9

  The Touchpad and TrackPoint 11

Today’s Interface - Good Versus Bad 12

Percent of Code 16

Voice User Interface – Automatic Speed Recognition and Text-To-Speech 17

  Alexa 19

  Automatic Speech Recognition and the Law 25

Zero User Interfaces 26

User Interfaces Assist the Disabled 29

Future 32

Conclusion 35

Footnotes 36

Disclaimer: The views, processes or methodologies published in this article are those of the author. They do not necessarily reflect Dell EMC’s views, processes or methodologies.

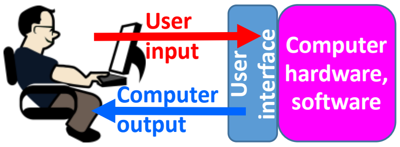
Human-Computer Interaction is a field of study focused on how we work with computers. Fundamental to our success with a system is its user interface. Over the history of computers, we have seen machines evolve from vacuum tubes, plug boards, and transistors, and user interfaces progress from punch cards, graphics, voice and neural user interfaces. Marked by rapid advances in technology and documented by famous principles such as Moore's law, cheaper and faster computers have attracted a less technically sophisticated yet ever-increasing user population. It has reached a point where the marvels we carry in our pocket have far more compute power and are easier to use than machines of just a decade earlier. A lot of credit for this success is due to the evolution of the user interface. For example, the blind are being assisted by a haptic zero user interface that leverage computer technology to get them safely to their destination, and quadriplegics who experience a better quality of life through a voice user interface. This paper is about the future of user interfaces and how they will help us interact with our computers.

"Moore's law is the observation that the number of transistors in a dense integrated circuit doubles approximately every two years."
https://en.wikipedia.org/wiki/Moore's_law

Introduction

Every day we interact with hundreds of things that share a common thread - they depend on an interface. Wake in the morning and slide a switch to turn off the alarm, twist a doorknob to leave your bedroom, rotate a shower faucet, switch on a coffee maker, press a remote control button to watch the morning news, and more. Your day is filled with countless interfaces. Some are intuitive and we hardly give them a thought, while others are taxing and may make you feel uncomfortable. Turning on a light using its switch interface is something you learned early on and it took little instruction, yet a VCR's menus and remote control drove older people crazy.

Successful interactions depend on the interface that controls the device. When a driver starts a car's engine, presses the gas pedal, and turns the wheel, the car responds by starting, going faster, and changing direction.

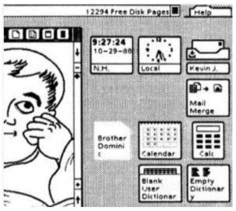


Many computer users believe its interface ***is*** the device. They perceive their interactions with keyboard, mouse, and screen ***are*** the computer when it is just the machine's User Interface (UI).

What is an Interface?

An interface sits between you and technology, and nearly every technology has one. Yet when we say the word interface, we inevitably think of the UI between a user and a computer, smartphone, tablet, or similar device. They can be physical devices such as keyboards, mice, touchscreens, and virtual objects such as screen icons and menus, voice-driven natural language assistants, gesture recognition devices, and more.

Text-based UIs ushered in the Graphical User Interface (GUI) we use on display screens, and navigate with mice, stylus, and fingers. GUIs came of age with Xerox PARC's 1970's desktop metaphor to the left, PCs, the internet, and websites.[1] Today, UIs are rooted in our user experience and include colors, device or screen appearance, content, responsiveness, and more. We lose interest or even make mistakes with slow or complex UIs, so a great deal of effort goes into improving them.



Exploring the future of UIs requires we understand their past. So let's start with a brief history of computer user interfaces, and see where the UI journey will take us in 2018 and beyond.

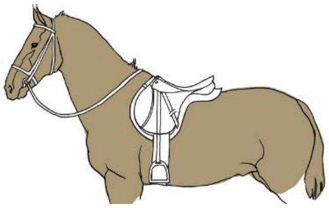
The Evolution of the User Interface

Interface Evolution

Interfaces have been around since the dawn of mankind, and some are more memorable than others. Let's take a look at some of them.

Horse

About 5,000-6,000 years ago, the first interface mankind used to control a horse was a rope around its neck or nose.[2] The rope interface evolved into a bit that was placed in the horse's mouth and was eventually made of metal along with a bridle interface to guide the horse sometime between the 14th and 8th century BC.[3,4] The saddle was likely introduced around 700 BC.[5] Together, the bridle and saddle comprise a rider's "horse interface." The bridle interface controls the horse's direction and pace, while the saddle allows the horse to better shoulder the rider's weight and allows the rider to help control the horse using the stirrups.



Car

Henry Ford's first car was called the Quadricycle. With four bicycle tires, a boat-like tiller directional arm to steer it, no brakes and no reverse gear, it weighed 500 pounds, used a crude 4 horsepower engine and had a bell on the front to warn people and horses that it was coming down the road.[6,7] Its user interface could have gone horribly wrong, such as using a horse bridle or ropes attached to the wheel's axels to steer it, but fortunately, over time, the car and its interfaces grew and matured, and now we have a steering wheel, power brakes, durable tires, and other standard features.

Elevator

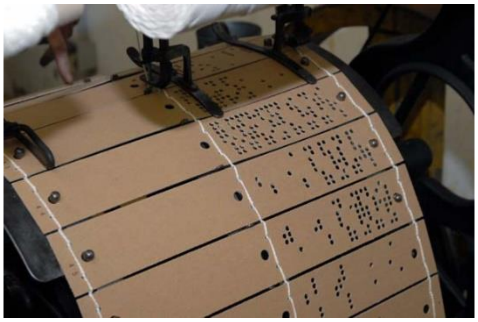
With a history dating back to the Roman Empire, the elevator was the epitome of modern convenience and an essential part of modern hi-rise apartment living. Elisha Otis is regarded as the father of the elevator and is known for creating a safety break that prevented it from crashing if its cable broke.[8] His Otis Elevator, up until the 1960's, had no end-user interface and required a semi-skilled "elevator operator" to manually open and close the doors. The worker remotely controlled the elevator's motor using an "up" and "down" lever pictured to the left to level a car with a floor as well as set the car's speed.

These days, elevators employ computer automation, allowing us access to its complex controls and programming using a simple interface of buttons without the need for a dedicated operator. Modern elevators algorithmically optimize its travel to reduce passenger wait times, handle the automatic sensors that signal it is safe to close doors, move the car to the exact location where it is safe to open the doors, reduce energy usage through regenerative braking, and respond to emergencies in a predetermined manner, all without any rider interaction.

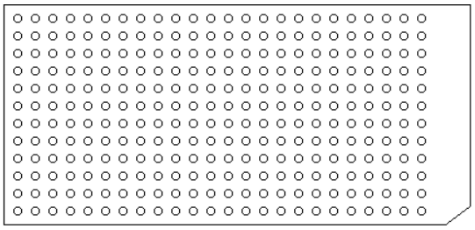
The Evolution of Computer Interfaces

The advent of the computer necessitated a user interface. Decades ago, computing power was expensive and programming focused on optimal designs rather than the UI. Punch cards and line printers were common, but hardly the things we dream of when we think of user interfaces. In this modern age where compute power tends to double yearly following Moore's law, we often assume that UI innovation also moved at the same breakneck-speed, but that wasn't always the case.[9] It took decades for computer screens, keyboards, and mice to evolve. We now have the world of smartphones and voice assistants, and the promise of their help to get us through our day. With future user interfaces, even grander things are in store.

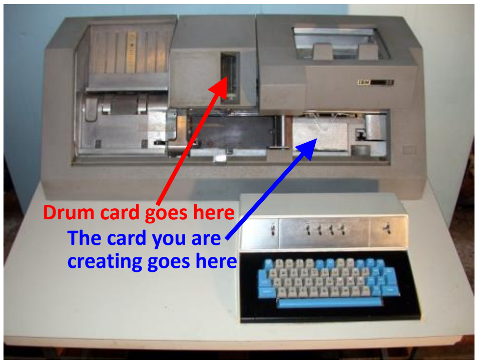
The Punch Card



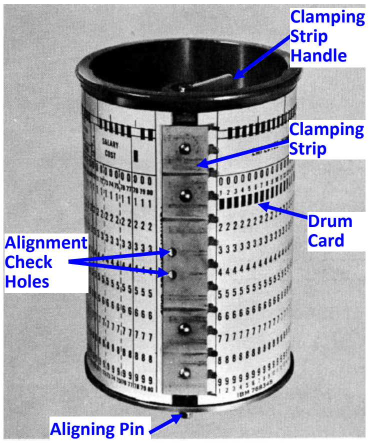
When electronic computing arrived, they leveraged early user interfaces such as paper punch cards and teletypewriters. The earliest use of punch cards was in the Jacquard loom of 1804 shown to the left.[10] The loom's UI consisted of a loop of cards to weave patterns, with each card controlling individual yarns in one row of the pattern.



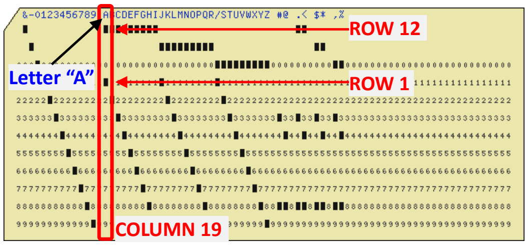
Early tabulation equipment, a precursor to the computer, was created by Herman Hollerith to help process the 1890 U.S. Census. His company made a machine that could punch 12 holes in 24 card columns, like this one to the left, and another machine that collected the data represented by those holes.[11,12] Hollerith's company became International Business Machines in 1924. Punched holes gave way to rectangles to get more data on a card, and changed its dimensions to the size of an old dollar bill. Early computers, such as the ENIAC, used punch cards as their sole user interface, not only for system input but also for output by printing upper-case letters and numbers across the top of the card (there were no lower-case letters).



Programmers used a rigid UI to work with these early computers, communicating through punch cards in languages like COBOL and FORTRAN. The cards were prepared with keypunch machines like this 1964 IBM model 029 to the left.[13] The machines were tools of the trade if you wanted to get a program or data into a computer.



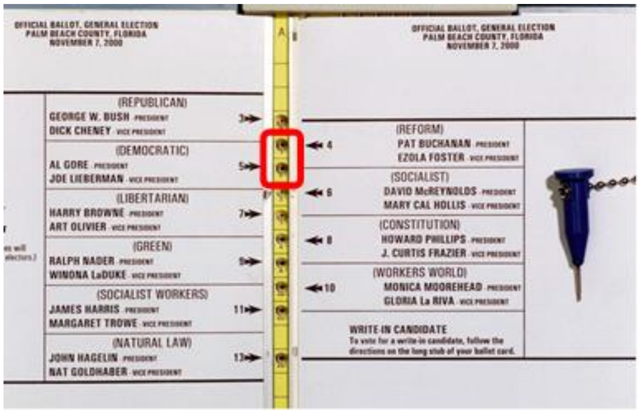
To make this user interface easier to work with, you used a shortcut template "drum card" shown to the right. It auto-positioned the blank card you were creating to a preset column or duplicated specific data from the previous card. The drum card rotated in step with your work and when twelve springy metal feelers sensed a punched rectangle, a relay instructed the keypunch machine to take an action.



From a UI perspective, hole placement had special meaning. Rather than force the user to memorize each pattern, the keypunch’s circuits signaled a predefined pattern of rectangles needed for each keypress. For example, the letter “**A**” was a punch in rows **12** and **1** in column **19** of this card.[14] Some of the fields of the card contained metadata, such as a card sequence in columns 1-5 of FORTRAN cards, a field in column 6 for multi-card continuations, and some simple identification in columns 73-80. COBOL had a different layout.

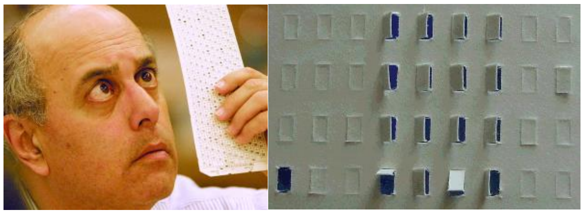
Source code and data punch cards were submitted to the computer’s card reader in a batch, while some machines used a punched paper tape, all to feed information into the computer’s memory as a stored program or data.[15]

Hanging chad


OFFICIAL BALLOT, GENERAL ELECTION
PALM BEACH COUNTY, FLORIDA
NOVEMBER 7, 2000
(REPUBLICAN)
GEORGE W. BUSH -PRESIDENT
DICK CHENEY -VICE PRESIDENT
3
(DEMOCRATIC)
AL GORE -PRESIDENT
JOE LIEBERMAN -VICE PRESIDENT
5
(LIBERTARIAN)
HARRY BROWNE -PRESIDENT
ART OLIVIER -VICE PRESIDENT
7
(GREEN)
RALPH NADER -PRESIDENT
WINONA LaDUKE -VICE PRESIDENT
9
(SOCIALIST WORKERS)
JAMES HARRIS -PRESIDENT
MARGARET TROWE -VICE PRESIDENT
11
(NATURAL LAW)
JOHN HAGELIN -PRESIDENT
NAT GOLDHABER -VICE PRESIDENT
13
OFFICIAL BALLOT, GENERAL ELECTION
PALM BEACH COUNTY, FLORIDA
NOVEMBER 7, 2000
4
(REFORM)
PAT BUCHANAN -PRESIDENT
EZOLA FOSTER -VICE PRESIDENT
6
(SOCIALIST)
DAVID McREYNOLDS -PRESIDENT
MARY CAL HOLLIS -VICE PRESIDENT
8
(CONSTITUTION)
HOWARD PHILLIPS -PRESIDENT
J. CURTIS FRAZIER -VICE PRESIDENT
10
(WORKERS WORLD)
MONICA MOOREHEAD -PRESIDENT
GLORIA La RIVA -VICE PRESIDENT
WRITE-IN CANDIDATE
To vote for a write-in candidate, follow the directions on the long stub of your ballot card.

A punch card user interface discussion would be incomplete without mentioning its role in one of the biggest fiascos in history. In the 2000 U.S. Presidential election, Florida’s Palm Beach County used a punch card ballot. On the left, a voter **punched a hole** next to their choices using a punch tool. It was then tabulated in in a special card reader.

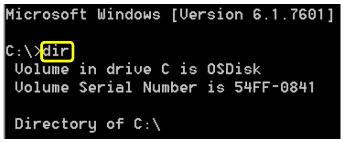
While simple in concept, the punch card UI did not prevent voter confusion or errors. Some ignored the black arrow and punched holes for both the Presidential and Vice-Presidential candidate (in the U.S., we only vote for the President), while some voted by mistake for both candidates Gore and Buchanan and others intended to vote for Gore but punched the Buchanan hole. Nothing in this design would help the voter catch their mistake.



Many did not fully punch out the hole, creating attached fragments called “hanging chad.” Chad repeatedly fouled up automatic tabulations. In this case, people correctly used the card UI but the tabulator couldn’t properly process their vote. Voting recounts went on for weeks with examiners supplying their chad interpretations of the voter’s intent. George Bush eventually won the

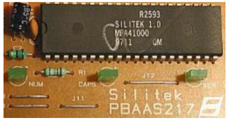
Florida vote by 537 votes over Al Gore. This example of a bad UI may have changed history and thankfully it helped usher out punch card voting systems.

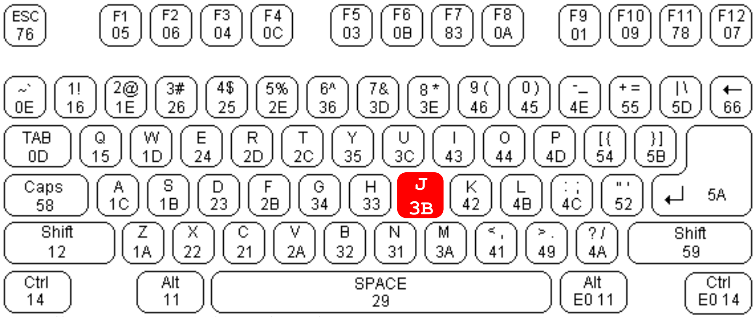
The Keyboard



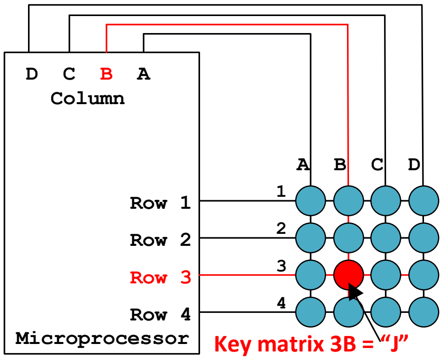
```
Microsoft Windows [Version 6.1.7601]

C:\>dir
 Volume in drive C is OSDisk
 Volume Serial Number is 54FF-0841

 Directory of C:\
```

While keyboards were required for punch cards, they really became common when the PC burst onto the scene in the early 1980’s. Users back then used an MS-DOS Command Line Interface (CLI) to enter commands like “**dir**” to get a list of files and folders. Coming in all shapes and sizes, the keyboard UI used dedicated trace circuits that touched with a keypress. Each key is mapped into a matrix where each intersection had a unique binary code. A keyboard microprocessor shown to the right scans the matrix and sends the appropriate hexadecimal scan code through the keyboard connector or USB port to the motherboard. In this example, pressing “**J**” is picked up as “**3B**” by the microprocessor, which forwards the scan code to the keyboard circuit on the PC’s motherboard. This circuit generates a hardware interrupt that calls a keyboard service routine into action. The routine processes the scan code by putting a two-byte code (the main and auxiliary byte) into a keyboard buffer area in RAM.

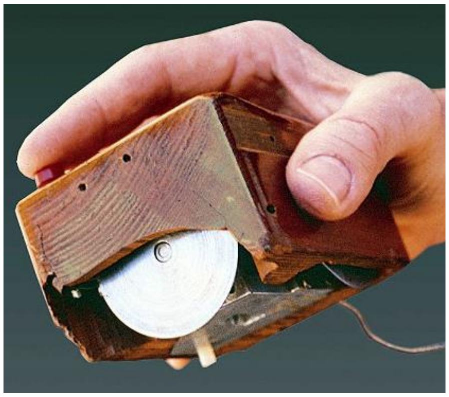




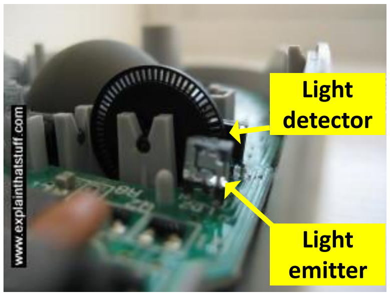
http://www.philipstorr.id.au/pcbook/book3/scancode.htm
http://pcbheaven.com/wikipages/How_Key_Matrices_Works/



The Mouse



Few user interface innovations have had more impact on human-computer interaction than the computer mouse, a pointing device that helped launch the desktop metaphor. While working at Xerox PARC in 1964, Douglas Engelbart built the mouse from a piece of wood and two metal wheels that moved when rolled on a surface.[16] Eight years later, the idea was tweaked to use a rubber ball in the mouse instead of wheels. It was also made ergonomic.

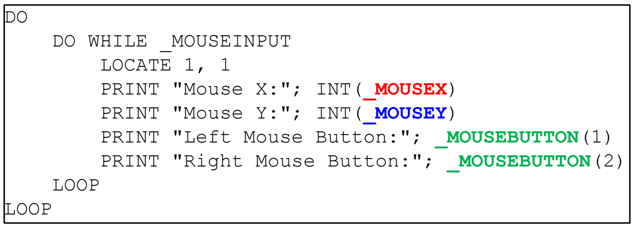


The movement of a mouse ball against a desk turned directional slotted wheels, allowing light to shine through its spokes and register on a photocell. Changes in direction of each slotted wheel break a light beam. These disruptions are counted as X and Y position changes, giving a relative distance to the mouse controller, and instruct your display's mouse pointer to move.[17]

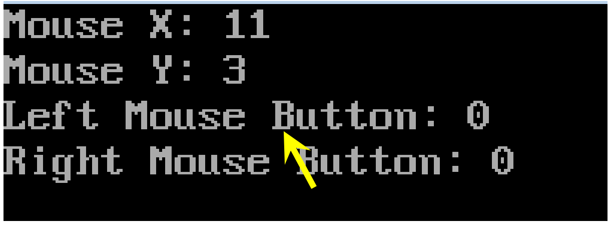
Around 1980, the mechanical ball was replaced by an optical sensor. A red LED reflected off a flat surface back to the mouse's photocell. Mouse movement changes the pattern of light that the photocell receives, and it is converted into X and Y movement.

Mice also have a wheel on top to change the X and Y coordinates, and buttons that send a "selection" signal to the controller. There are many types of mice such as wired and wireless, and all sorts of button arrangements.

Below is the code fragment that illustrates the simplicity of integrating a mouse UI into a computer program.[18] The code relies on built-in functions that detect mouse event messages such as a button press or release, or when the mouse changes its X or Y position. Messages are queued. This code fragment uses the **_MOUSEX** and **_MOUSEY** functions which returns the X and Y coordinates of the mouse, and the **_MOUSEBUTTON**(1-3) function which returns the state of the button (-1 if the button is pressed, 0 if it is not).[19] In this example, the mouse is pointing below the word "Button" which the program senses as column 11 and row 3.

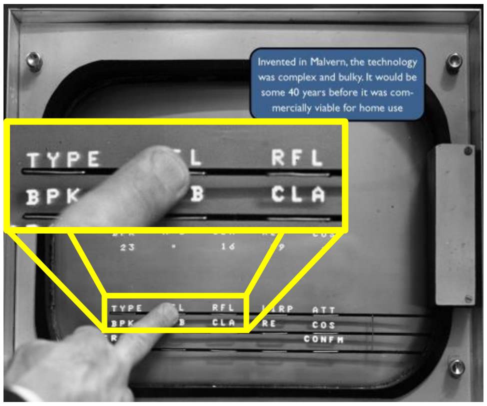


```
DO
    DO WHILE _MOUSEINPUT
        LOCATE 1, 1
        PRINT "Mouse X:"; INT(_MOUSEX)
        PRINT "Mouse Y:"; INT(_MOUSEY)
        PRINT "Left Mouse Button:"; _MOUSEBUTTON(1)
        PRINT "Right Mouse Button:"; _MOUSEBUTTON(2)
    LOOP
LOOP
```

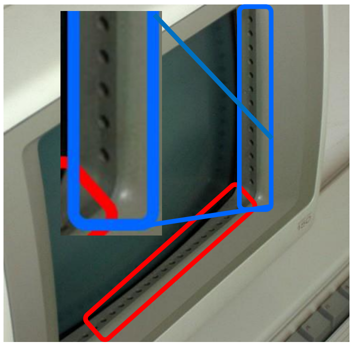


```
Mouse X: 11
Mouse Y: 3
Left Mouse Button: 0
Right Mouse Button: 0
```

The Touchscreen


Invented in Malvern, the technology was complex and bulky. It would be some 40 years before it was commercially viable for home use
TYPE
RFL
BPK
B
CLA
23
16
ATT
RE
COS
CONFM

Before computer mice became popular, and at a time when most computer interactions involved punch cards and paper printouts, the touchscreen represented a more natural way of interacting with a display screen. Developed in 1965 by Eric Johnson of England's Royal Radar Establishment, his user interface consisted of copper wires attached to the bottom of a screen.[20] Used primarily to select a specific range of choices using the alphanumeric 0-9 and A-Z, a user touched a labeled part of a display screen. The underlying software translated the X-Y coordinates of the pressed wire into the user's selection. In the case of Mr. Johnson and radar systems, aircraft had three-letter designations, so it was an effective way for an air traffic controller to select a specific flight.



Some touchscreen UIs later adopted infrared light emitters and transducers embedded in a computer screen bezel. They could interpret a finger blocking a light wave as it touched the screen such as with this 1983 HP-150 PC.[21] Housed in tiny holes alongside the screen as shown by the **red** and **blue** highlighted area, a finger touch is communicated to the underlying MS-DOS program as the X-Y horizontal and vertical location of where the user pointed.

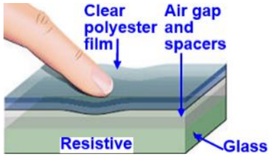
While touchscreen technology suffers from a lack of precision, certainly as compared to computer mice, this user interface has found widespread use where large block graphical choices are presented and the use of a mouse is impractical. For example, automated teller machines (ATM), airport check-in kiosks, and some car displays leverage a UI that limits on-screen choices and moves a user through a defined sequence to complete a transaction.

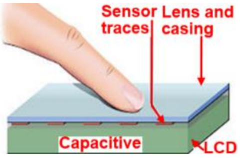
While the GUI has evolved from a keyboard and mouse to include touchscreen desktops and laptops, most of the touchscreen development has been targeted at smartphones and tablets and is part of a movement towards a Natural User Interface (NUI). The goal of a NUI is to operate "…through intuitive actions related to natural, everyday human behavior."[22] According to Microsoft researcher Bill Buxton, NUIs "exploit skills that we have acquired through a lifetime of living in the world, which minimizes the cognitive load and therefore minimizes the distraction".[23]

A more natural user interface can be accomplished through multi-touch technology. Multi-touch leverages two or more screen touchpoints in contrast to a mouse which handles a single point of screen interaction. While a mouse is ingrained into our lives, controlling a single point with our multi-jointed hand may limit productivity.

The idea behind multi-touch is simple – allow us to convey more information to the computer in a shorter time and in a more natural way. Multi-touch will not be ten times faster than using a mouse, just like typing with ten fingers is not ten times faster than using a single finger, but depending on the UI, we can increase productivity and have a higher rate of information input.

The iPhone is an NUI device with both a good and bad multi-touch user interface. Some iPhone gestures are natural, like swiping from left to right with your finger to turn a page resembles how you would turn the page of a book. Others are far less obvious and require learning, such as taking an iPhone screenshot as shown to the left which requires the multi-touch of your right hand on the power button while your left hand presses the Home button before releasing them both.





The basic types of NUI touchscreens in use today are **capacitive** and **resistive**.[24] The 2007 iPhone and touch-screen desktops are examples of **capacitive** devices that use your fingertip's skin as a conductor. These surfaces use coordinate grids of embedded electrodes, and touching it completes a circuit. An iPhone microcontroller passes the coordinates and other gesture-interpretations to the iOS operating system and ultimately to the running app. A major **capacitive** drawback occurs in winter or in industrial scenarios when you are wearing non-conductive gloves - an iPhone's conductive touchscreen would not work.

Resistive touchscreens, sometimes called pressure sensitive screens, use the pressure of a stylus or a pointed object to compress a flexible conducting plastic against conducting glass separated by a membrane. An ATM touchscreen may be **resistive** given they can be found outdoors and used in winter time. Some screens blend **capacitive** and **resistive** approaches. Either method senses the electrical change, allowing the software to indicate cursor movement or a press. In many ways, this technology borrows heavily from the computer mouse.

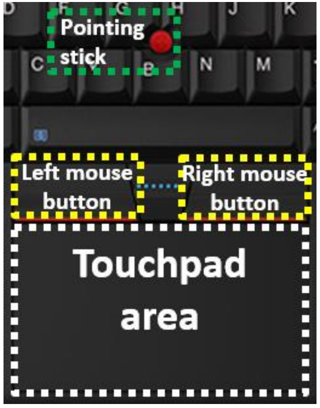
iOS Function	Description
func touchesMoved(Set<UITouch>, with: UIEvent?)	Tells the responder when one or more touches associated with an event changed.
func touchesEnded(Set<UITouch>, with: UIEvent?)	Tells the responder when one or more fingers are raised from a view or window.
func touchesCancelled(Set<UITouch>, with: UIEvent?)	Tells the responder when a system event (such as a system alert) cancels a touch sequence.
func touchesEstimatedPropertiesUpdated(Set<UITouch>)	Tells the responder that updated values were received for previously estimated properties or that an update is no longer expected.
func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?)	Tells this object that one or more new touches occurred in a view or window.

If you were writing a program to capture single iOS pointing events, you might use these Apple developer tools on the left.[25]

The Touchpad and TrackPoint

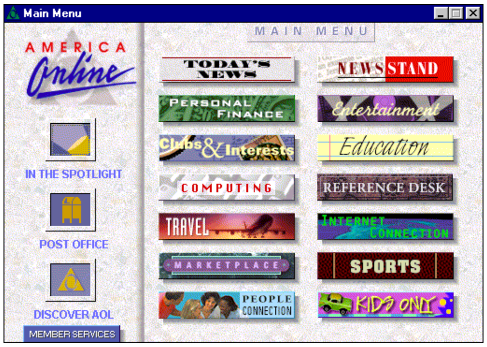
Sharing some touchscreen properties, most NUI laptop touchpads (pad) use capacitance to track finger movement. From a UI perspective, a pad is like a mouse, but with a smaller working area. They use physical left and right "mouse" buttons or simulate buttons with a tap on the pad.

Some pads have a scroll area to the side to simulate a mouse wheel. Modern pads use a matrix of conductors below its surface separated by an insulator and a grid at right angles on top. A finger touch changes the capacitance. These changes and the grid intersection is passed to the operating system and appears as mouse activity to a program, adhering to the _MOUSEX and _MOUSEY APIs discussed earlier. Laptops can also have a pointing stick like this **red** one on IBM's ThinkPad to the left. Situated between keys, it behaves like a joystick and appears to a program as a mouse.



Today's Interface - Good Versus Bad

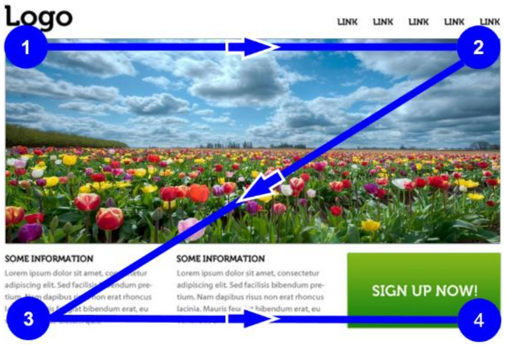
Interfaces are intermediaries that shield us from the underlying complexities of what we want to do from how the item needs to operate. In the computerized world, those complexities can be the underlying programs, operating systems, and networks. For example, when AOL, Prodigy, and CompuServe were popular, they all tried to shield us from some of the underlying intricacies through their GUI. Nowadays, Firefox, Edge, Safari, and Chrome browsers have user interfaces that hide the underlying HTML, style sheets, and scripts.



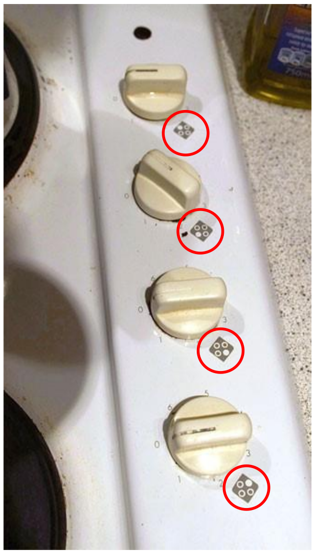
Web and app designers strive to create user interfaces that prevent us from making mistakes, improve productivity, ensure smooth operations, and shield us from inefficient, confusing or unusable products and systems. They employ design and implementation techniques that fall under the scope of human-computer interaction. Many of the standards governing this field of study are contained in ISO 9241 "Ergonomics of human-system interaction".[26]

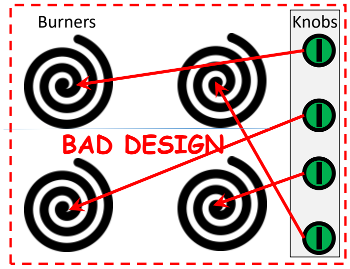
The benefits of a good UI are tangible and measurable. In the business world, good interfaces lead to higher morale and job satisfaction, and lower training costs and staff turnover, all leading to lower operational costs.

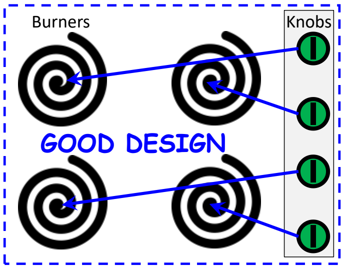
From a user's perspective, the UI is a gateway into their computer and represents easy access to the intricacies of its underlying hardware, software, and networking. For example, when you enter an elevator, its interface works with elevator logic to close the door, get you to your floor at an acceptable speed, process multiple floor requests, adjust the floor indicator, open the door, and dispatch the car to waiting passengers.



Based on reading patterns, successful web pages leverage a UI shaped like a letter "**Z**".[27] The eye tends to travel from the ① top left of a graphical screen across to the right②, then diagonally to the bottom left③, and finishing on the bottom right④. This "**Z**" shape follows how westerners mostly read left-to-right and are often associated with a good user experience.

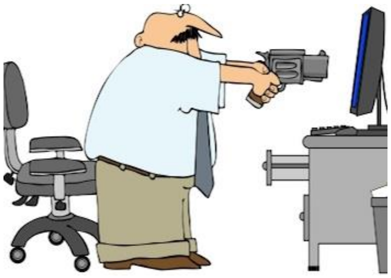




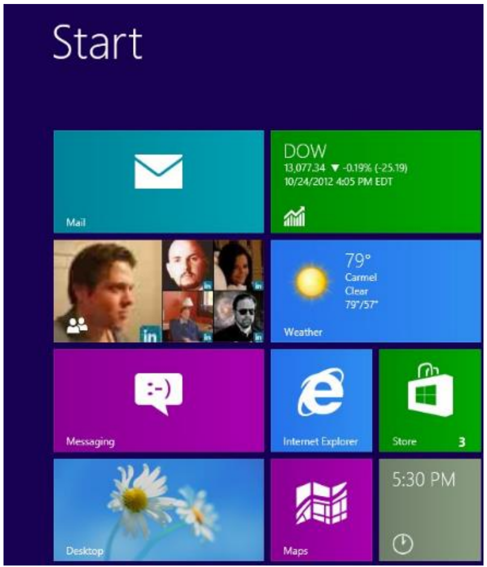


While it can be difficult to judge how good an interface is, it is usually easy to spot a bad one. On the left, the stove's burner and knob relationship are not intuitive.[28] Each knob has a diagram showing the burner it controls, but the order is confusing, leading the user to turn the wrong knob. With the 2nd knob from the top, the user expects to turn on the top-right burner, but it controls the bottom-left one, which could have dire consequences. The diagram to the right illustrates this **bad design** and a suggestion for what might be a **good design**.

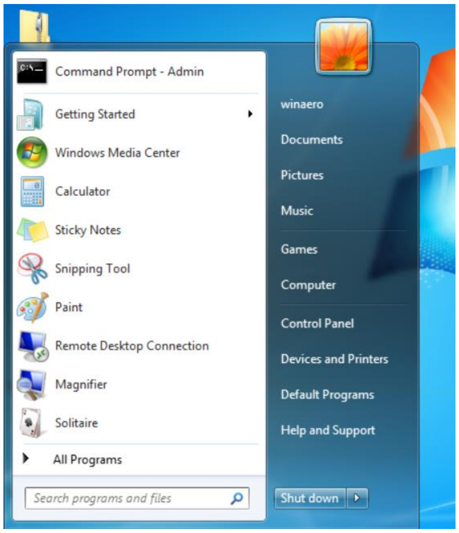
Another bad interface was involved in the deaths of 290 passengers on an IranAir A300 passenger jet. In 1988, a U.S. Navy guided missile cruiser with a state-of-the-art Aegis Weapon System gave its captain incomplete information, resulting in the A300's accidental destruction. The captain believed an Iranian F-14 was about to attack his ship and fired radar-guided missiles. While there were many factors that led to this tragedy, two key Aegis UI elements were at fault. The Aegis operators mistakenly told the captain that 1) the jet was descending to attack when it was ascending, and 2) the jet's Identification, Friend or Foe (IFF) code showed it was a "Mode II" military plane and not a "Mode III" civilian plane. Aegis is very complex and uses three displays. To get a bogey's speed, range, and altitude, the operator finds it on a small 12" screen and manually calculates the rate of change. As for the IFF error, the system correctly identified the A300 as Mode III, but the operator's trackball cursor pointed to a parked F-14 at the same airport the A300 took off from.[29,30]

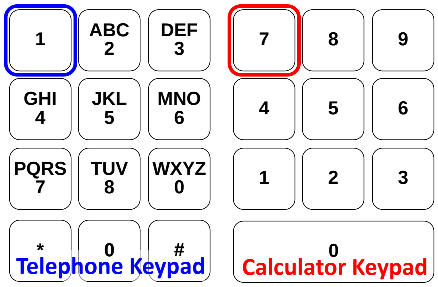


A poor UI can add to employee stress which can cost a business as much as $40,000 per person/year in lost productivity.[31] When the user interface fails to shield us from the underlying complexity or rises to a level beyond which we can tolerate, "computer rage" can occur. Computer rage happens when the machine doesn't do what you want it to do. We've all done it – cursing, yelling, and slapping the side of our helpless machine. It can happen when it runs slowly, loses the document you've been slaving on, or other computer problems. The user interface is not totally at fault, but it likely could have helped $^{1}/_{3}$ of Americans who lashed out at their computers when they failed to do what the user expected.[32] In some cases, a UI gauge displaying a realistic timeframe for an activity to occur, better error messages, or even a simple popup to remind us that we entered a full name in a field that is expecting a credit card number.

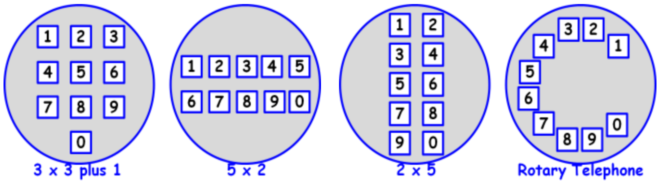


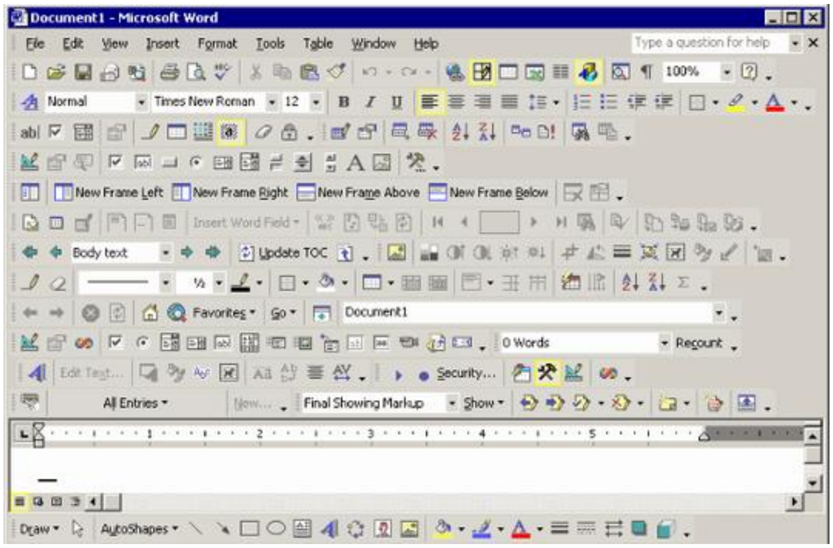
From a GUI standpoint, Windows 8 on the left is another example of a poor UI. It is perhaps one of the biggest design disasters in modern computing history and probably cost Microsoft and the entire PC ecosystem a considerable amount of money. In 2012, they introduced a phone-inspired touch-oriented tile interface to PC users who were accustomed to using a mouse and keyboard. Windows 8 forced users to abandon their long-standing "Start" button and menus, ending the engrained UI familiarity shown to the right. It mandated they learn a new way of getting tasks done without any perceived productivity gains. The backlash was quick and fierce. Confused users doomed the operating system, and some resorted to typing in the application name, like Word, just to start it because they couldn't find the right tile to press – a "Where's Waldo" moment. Journalists bashed Windows 8, resulting in the adoption rate reaching only 8% while 46% of users remained on Windows 7.[33] The reception was so poor that Microsoft rushed Windows 8.1 to market a year later with a Start menu button and other fixes.





One of the obvious UI tradeoffs is the difference between **telephone** and **calculator** keypads. On the left, the **telephone** has the numeral “**1**” at the top left and the **calculator** has a “**7**” in that spot. Can both pad interfaces be highly successful if they are opposite to each other? While some of this is still unknown, a 1960 Bell Labs study of telephone push buttons concluded a 3 x 3 matrix with zero at the bottom was better than a 5 x 2 or 2 x 5 matrix, or even a circular rotary phone pattern.[34,35] They also found from a human factors interface standpoint that 1-2-3 across the top row was easier because of the letter sequence that corresponded to the beginning of the numbers. Calculator pads came about in 1914 and resembled cash registers of its time.[36]





While there is little agreement on exactly what makes a good user interface, as we have seen, we can certainly agree on a bad one when we see it. While we would never activate all of Microsoft Word’s menus, if we did, it would be overwhelming with little room left to do any actual typing in the document area.

The following user interface design concepts are probably good to include.[37]

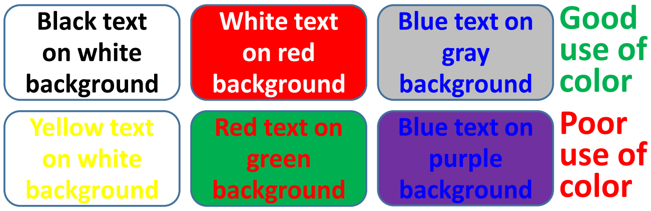
- **Clear** – a UI needs to be easy to use and obvious. For example, present the user with a sorted list if it is appropriate.
- **Concise** – clarity is important, but it should not be overly wordy.
- **Familiar** – people learn new concepts based on ones they already know.
- **Responsive** – user feedback is key. Rapidly acknowledging a GUI screen press.
- **Consistent** – visual interfaces in a series of screens should have the same “look and feel.” For example, Microsoft Word, Excel and PowerPoint have a similar appearance.
- **Attractive** – users may need to use the interface daily, so it should be nice-looking.
- **Efficient** – users should interact with complex applications in the simplest way possible. For example, parse their “first” and “last” name instead of using two fields.
- **Forgiving** – if the UI can’t prevent a user error, it should allow the user to correct it.

Percent of Code

Software developers face an uphill battle. User interfaces must accommodate a user's ability, expectations and experience levels. From a human factor perspective, the developer needs to assume a user cannot recall more than seven pieces of information, and design for error detection and correction to avoid stress and the mistakes it can cause, since once a user makes mistakes, frustration can cause them to make a lot more – a "catch-22." [38] In addition, each user is different, and some respond better to textual prompts while others prefer pictures and icons.

In the end, the UI represents a significant part of the developer's job. It commands the user's focus, so if not done well, the developer may unknowingly create a confusing program even though it uses the best algorithms, employs efficient code, etc. Studies have shown that 47-60% of the code is related to the UI, and for the UI to evolve and become end-user friendly, the code winds up getting even more complex.[39] For example, it takes a tremendous amount of code and CPU cycles to analyze a spoken user response compared to a "Yes/No" command line prompt. In addition, the underlying code must be tested, documented, have user help, language translations and more. UI design is generally iterative and central to the User Experience (UX), so it is common to redesign portions of the code after user-testing. Given the underlying complexity of technology, simple UIs must achieve the magic necessary to create a pleasant UX. The more successful the UI and UX, the more blurred the distinction between an interface and underlying technology.

The goal of a good UI is to be transparent, allowing users to focus on their work and not on the interface. Transparency can be as simple as a visual layout, effective use of colors, asking clearer dialog questions, making it faster, or reorganizing logic to involve the user in dialog only when necessary.[40] Good user interfaces also lower training support costs.

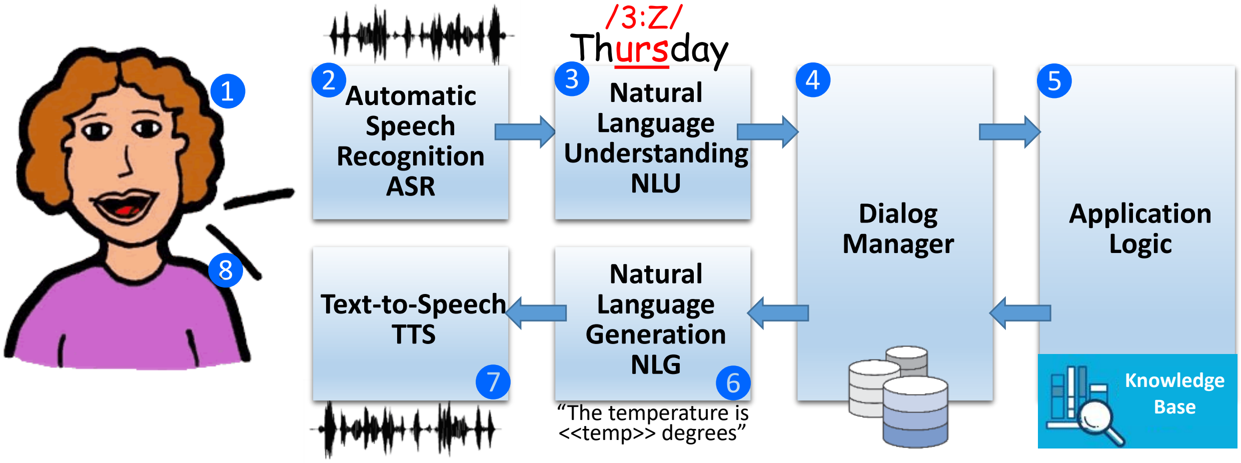

Black text on white background
White text on red background
Blue text on gray background
Good use of color
Yellow text on white background
Red text on green background
Blue text on purple background
Poor use of color

Compounding the complexity, the UI style for desktop browser-based traditional apps is different from touchscreen smartphone apps, and users have little tolerance for bad smartphone apps. In the U.S., the rejection rate is 19% and reaches 37% in China.[41] Application logic, such as the current weather or stock performance, cannot fundamentally change, so what is left is the user interface and user experience.

Voice User Interface – Automatic Speed Recognition and Text-To-Speech

The purpose of speech recognition and a Voice User Interface (VUI) is to support and enhance the interaction between man and machine. It has become an almost natural and intuitive form of human-machine interaction. The spoken language represents one of mankind's most powerful inventions and has allowed us to evolve our ideas rapidly over the last 5,000 years.[42] Talking to a machine is an idea that has been floated for many years, and many remember the scene from "*Star Trek 4: The Voyage Home*" when Mr. Scott, in search of information on transparent aluminum, picks up an Apple mouse and says "Hello computer?"[43] These days, we can talk to our cars, smartphones, banking applications and more. We talk faster than we type and convey our emotions in every conversation.

A VUI leverages natural language to take the man-machine interface one step closer to the goal of being seamless. They generally follow the steps depicted below.[44]


①
② Automatic Speech Recognition ASR
/3:Z/
Thursday
③ Natural Language Understanding NLU
④ Dialog Manager
⑤ Application Logic
Knowledge Base
⑥ Natural Language Generation NLG
"The temperature is <<temp>> degrees"
⑦ Text-to-Speech TTS
⑧

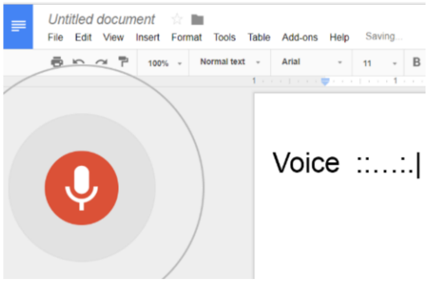
① Activated using a keyword such as "Alexa."
② Automatic Speech Recognition converts sound into text.
③ Natural Language Understanding uses the text to understand what the user wants.
④ Dialog Manager dispatches the request to a program to provide the "answer" using action and attribute pairs, passing it back to the Natural Language Generation routine.[45]
⑤ Application Logic is expecting a transactional query and responds to the dialog.
⑥ Natural Language Generation produces language by packaging dialog components.
⑦ Text-to-Speech maps the units of language to a pre-recorded database of sound waves.
⑧ The user hears the response.

One of the early natural language programs was ELIZA, a 1960's endeavor that allowed you to converse with a "make-believe" therapist.[46] The conversation bot ("chatbot"), designed to simulate a human therapist conversation, went something like this.[47]

```
ELIZA: Is something troubling you?
YOU:   I am having problems with my girlfriend.
ELIZA: Why do you say your girlfriend?
YOU:   Because I have been with her a long time and I love her.
ELIZA: Can you elaborate on that?
YOU:   I think she wants to leave me.
ELIZA: Do you say you think she wants to leave you for some special reason?
YOU:   I think she met someone else.
ELIZA: That's quite interesting.
```

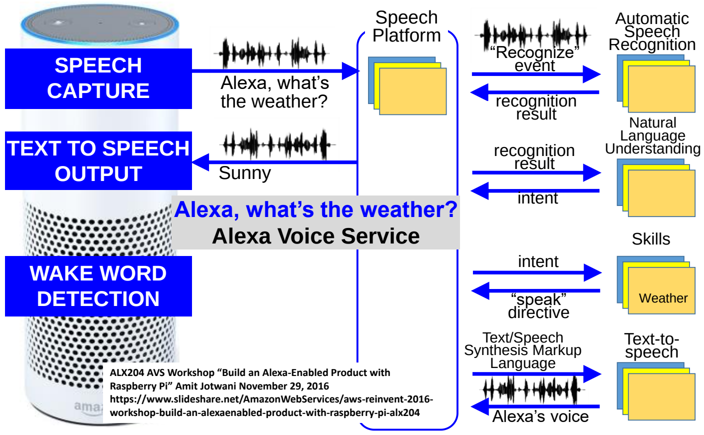
In 1952, engineers from Bell Labs came up with the "Audrey" system that could understand the spoken digits.[48] In 1985, Ray Kurzweil developed software that could recognize 1,000 spoken words if … enough … space … was … put … between … spoken … words. In 1997, Dragon Systems marketed a product that could decipher 23,000 words of continuous speech and ushered in an era where a UI could incorporate speech comprehension.[49] It was error-prone and required a great deal of training to improve its accuracy beyond 90-95%, meaning one word in ten could be wrong. Early speech comprehension products never received universal acceptance. For instance, in the medical industry, getting a word wrong every sentence could have dire ramifications, and end users quickly tired with the "out-of-the-box" experience. Dragon required you to speak into a microphone for the computer to transcribe your words, understand your commands, and read a document or selection out loud. It marked a breakthrough for disabled individuals since it no longer required the use of your hands and fingers.[50]

If you would like to experiment with Automatic Speech Recognition (ASR), you might want to try using Google Docs under the Chrome browser. With your computer's microphone on, start a Google document. Under **Tools**, click **Voice Typing**, and a microphone box appears. Click on the microphone and anything you dictate is transcribed into your document. If you want to try text-to-speech (TTS), you can also use iSpeech Company's URL www.ispeech.org/ and test their interface, which can be embedded into your application.


Untitled document
File Edit View Insert Format Tools Table Add-ons Help Saving...
100%
Normal text
Arial
11
B
Voice

In 2018, products capable of voice interaction without repetitive user training include Amazon Alexa, Google Assistant, Apple Siri, and Microsoft Cortana, and some of them represent the end of typing, mouse-ing, tapping and swiping. While a lot of the conceptual underlying technology is the same for this class of products, let's see how the Alexa user interface works.

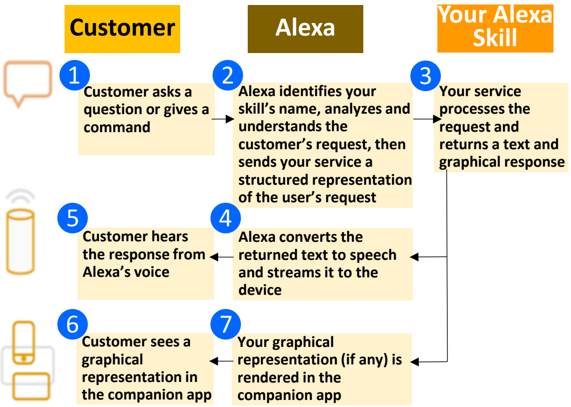
Alexa



ALX204 AVS Workshop “Build an Alexa-Enabled Product with Raspberry Pi” Amit Jotwani November 29, 2016 https://www.slideshare.net/AmazonWebServices/aws-reinvent-2016-workshop-build-an-alexaenabled-product-with-raspberry-pi-alx204

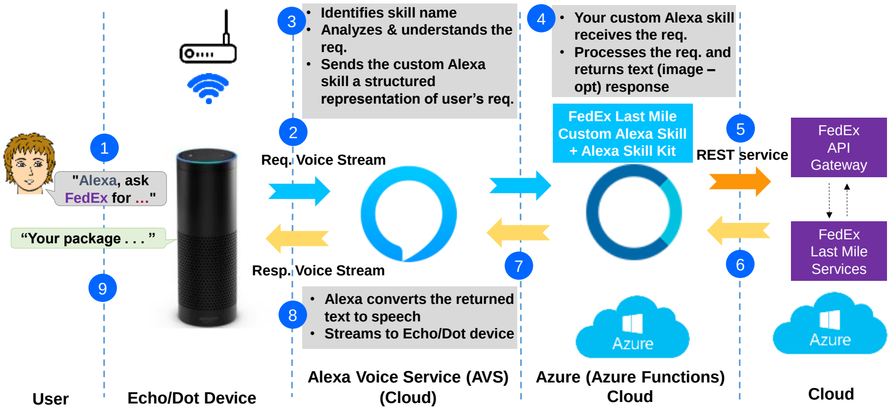
Alexa is Amazon’s virtual digital assistant that interfaces us with physical devices like Echo and Dot. It is always in passive listening mode and sends whatever it hears up to the Amazon Web Services (AWS) cloud once it detects the trigger word “Alexa?” It uses “skills” that are either provided by Amazon for very common requests such as questions about the weather, sports scores, news, and music, or third-party skills (apps) that can raise the house temperature or order a pizza. Questions and commands it can’t easily answer by voice may be responded to by text or pictures displayable on Amazon devices such as a Fire tablet, Fire-attached TV, or previously interfaced third-party products such as Philips Hue light bulbs, Nest thermostats, Schlage Connect door locks and other Alexa-connected products.

Developers add skills using the Lex Software Development Kit (SDK) to enhance the VUI.[51] The SDK provides a collection of RESTful APIs and tools that “…hear, understand, and process the customer’s spoken request…,” and allow for easy integration into existing applications.[52] Alexa’s UI is quite powerful, allowing developers to create new “intents” (an action a user wants Alexa to take for them), such as when a United Airlines customer wants to “book a flight” as part of their conversational interface for their application. Their chatbot includes an interactive dialog script played back to the user to fulfill this airline transaction, such as a question about the travel date and destination portion of this query. Under the covers, cloud-based Lex leverages ASR to convert speech to text and apply Natural Language Understanding (NLU) to recognize its intent.

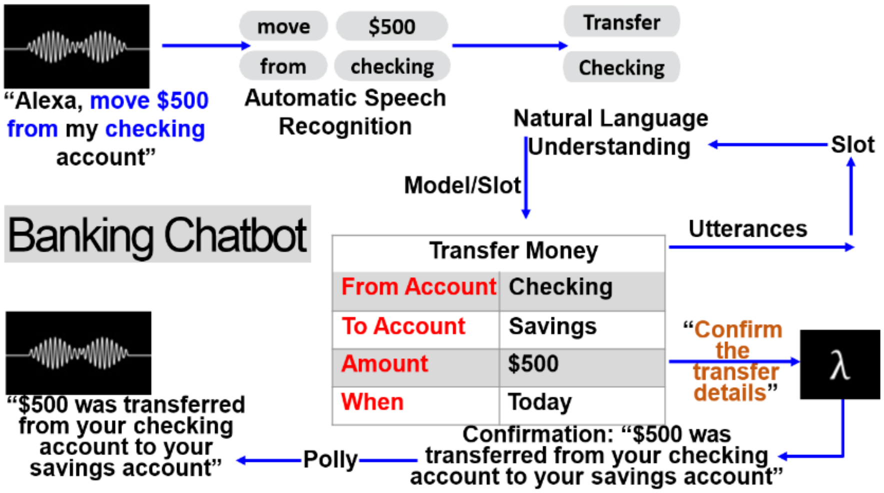

Customer
Alexa
Your Alexa Skill
1
Customer asks a question or gives a command
2
Alexa identifies your skill's name, analyzes and understands the customer's request, then sends your service a structured representation of the user's request
3
Your service processes the request and returns a text and graphical response
4
Alexa converts the returned text to speech and streams it to the device
5
Customer hears the response from Alexa's voice
6
Customer sees a graphical representation in the companion app
7
Your graphical representation (if any) is rendered in the companion app

https://developer.amazon.com/docs/custom-skills/understanding-custom-skills.html#components-of-a-custom-skill

For example, the “Starbucks Reorder Skill” knows to order your default beverage when you say "Alexa, order my Starbucks". Behind the scenes, a chatbot flow, such as the one to the left, ties in Starbucks’ custom skill using the Alexa APIs that reacts to their invocation name of “Starbucks”①.[53] When the user says “Alexa, open Starbucks,” a Starbucks routine walks the user through menu options, payment terms, and store pickup details.[54]


"Alexa, ask FedEx for ..."
"Your package . . . "
User
Echo/Dot Device
1
2
Req. Voice Stream
Resp. Voice Stream
3
Identifies skill name
Analyzes & understands the req.
Sends the custom Alexa skill a structured representation of user's req.
4
Your custom Alexa skill receives the req.
Processes the req. and returns text (image – opt) response
FedEx Last Mile Custom Alexa Skill + Alexa Skill Kit
5
REST service
6
7
8
Alexa converts the returned text to speech
Streams to Echo/Dot device
9
FedEx API Gateway
FedEx Last Mile Services
Azure
Azure
Alexa Voice Service (AVS) (Cloud)
Azure (Azure Functions) Cloud
Cloud

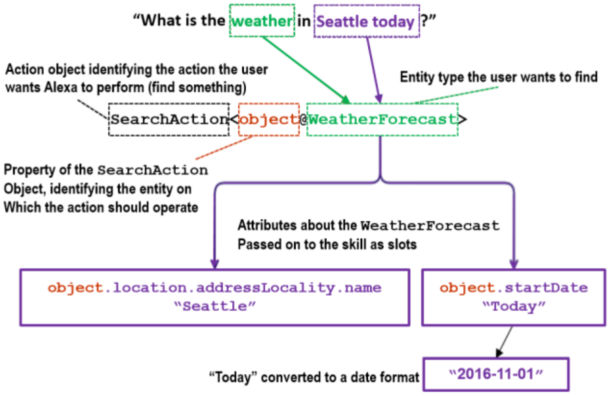
This FedEx example allows a customer to say “Alexa, prepare a shipment” or “Alexa, ask FedEx for a status on my package.”[55] The VUI chatbot (left) supports their “last mile” initiative using the Lex SDK of RESTful APIs to provide information that ordinarily requires a Windows browser, the FedEx URL to start the application, and the package ID to start a tracking query. With the Alexa VUI, the customer uses natural language to get the information. The Alexa Voice Service (AVS) runs in the AWS cloud, identifies the skill name and sends the request to the FedEx skill. The FedEx chatbot sends delivery information using the Alexa SDK to Polly to convert TTS in any of 47 voices and 24 languages and play it on the customer’s Echo in step ⑧.[56] Let’s take a deeper dive into how Alexa works.


“Alexa, move $500 from my checking account”
move
$500
from
checking
Automatic Speech Recognition
Transfer
Checking
Natural Language Understanding
Slot
Model/Slot
Utterances
Banking Chatbot
“Confirm the transfer details”
λ
“$500 was transferred from your checking account to your savings account”
Polly
Confirmation: “$500 was transferred from your checking account to your savings account”

Transfer Money	
From Account	Checking
To Account	Savings
Amount	$500
When	Today

In this hypothetical banking chatbot, the app is started by the phrase “**Alexa, move $500 from my checking account**”.[57] This banking skill was registered by the user in advance. ASR finds the words “**move**,” “**$500**,” “**from**,” and “**checking**” and passes it to the NLU module that starts the “**transfer from checking**” application. With the user’s account verified, the source and destination accounts are determined as is the amount of money to transfer and its effective date/time. With the transaction complete, the banking application passes the phrase “**Confirm the transfer details**” to Lambda.[58] Lambda is running the banking program, and it sends a more comprehensive message to Polly which announces the completed transfer to the user.

Alexa use speech recognition to parse each request or question using a library of natural language-like algorithms residing in AWS to respond to books, calendar, cinema showtimes, local search, music, video, weather, standard phrases (like stop, cancel, and help), and general intents that don't necessarily fall into a specific category.[59] Alexa's comprehension rate of human speech is about 95% - an improvement over earlier VUIs.[60]


"What is the weather in Seattle today?"
Action object identifying the action the user wants Alexa to perform (find something)
Entity type the user wants to find
SearchAction<object@WeatherForecast>
Property of the SearchAction Object, identifying the entity on Which the action should operate
Attributes about the WeatherForecast Passed on to the skill as slots
object.location.addressLocality.name "Seattle"
object.startDate "Today"
"Today" converted to a date format
"2016-11-01"

This intent pattern for weather follows a natural language construct composed of an action for Alexa to do "**What is the**," followed by the object of the action "**weather**," and the entity the object refers to "**Seattle today?**"[61] To reply, a skill starts a prearranged conversation to reach an outcome.

ASR takes unknown sound waves and converts it into signals that compare favorably to those in its database. While Alexa's VUI does a good job in its approach to Natural Language Understanding, it is not the same as full Natural Language Processing (NLP). Alexa is not powerful enough to fully extract the literal meaning of the words with enough pragmatics to infer a sentence's true meaning – i.e., it fails Alan Turing's test.[62,63]

> "A computer would deserve to be called intelligent if it could deceive a human into believing that it was human."
> - Alan Turing

Let's consider just how difficult NLP can be. In this sample sentence "**We saw her duck**," is the word "duck" a noun, as in a bird, or a verb, as to get down?[64]

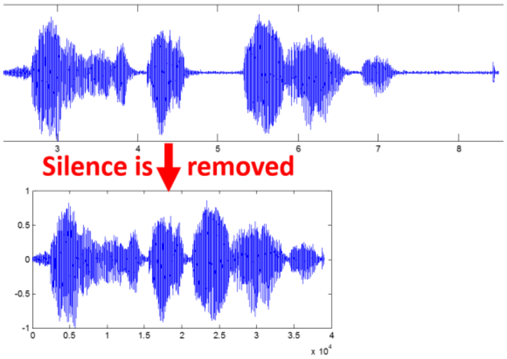
- That we viewed a certain bird that belonged to a female person?
- That we observed a female person perform the action of ducking?
- That someone named "Ms. We" spotted her pet?
- That we use a saw to cut her duck?

The sentence needs some context to know its true meaning.

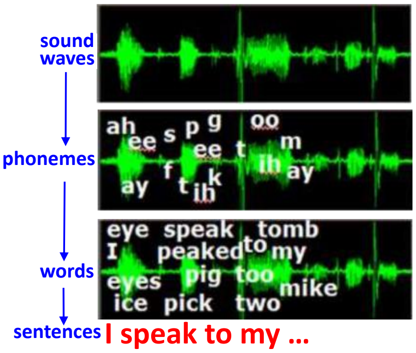
Have you ever corrected someone during a phone call because of poor pronunciation? NLP is even more difficult when you add different accents, speed, speech disorders, slang, dialects, locales, and languages. To illustrate these issues, consider the following six differences (out of many) between these simple American English and British English terms. There are also differences in spelling, such as color and colour, and prepositions such as "What are you doing on Christmas?" versus "What are you doing at Christmas?" Imagine the look on the faces of a U.S. Texan and a British Glaswegian as they each try to comprehend each other's accent.

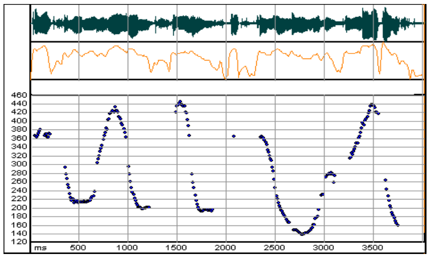
American English	British English
Apartment	Flat
(French) Fries	Chips
Drugstore	Chemist
Soccer	Football
Sneakers	Trainers
Eggplant	Aubergine

ASR listens to your words using a microphone (invented over 140 years ago by Emile Berliner) that converts sound waves into electrical signals.[65] Similar to text-based chatbots that scan for a carriage return/enter key (ASCII key code 13), or a question mark "?" (ASCII key code 63) to begin parsing a query, an ASR employs "endpoint detection" to determine when you start and stop speaking to know when it should process the sound waves into words. Care is taken when silence is detected because it can be a natural speech sequence such as saying the digits of a credit card number. Background noises are filtered out, volume normalization (not too high and not too low) performed for soft-spoken users, adjustments made for speech speeds, and excess silence removed as shown in this diagram.



This processed waveform shows the signal parsed into small segments, ranging from hundredths to thousands of a second, that match phonemes (pronounced fōnēm).[66,67] Phonemes are basic units of sound that form words and language (Spanish has 24 phonemes and English has 44 phonemes such as "b" as in build and "k" as in clap).[68,69] The world has 7,000 languages, each with a set of phonemes.[70]

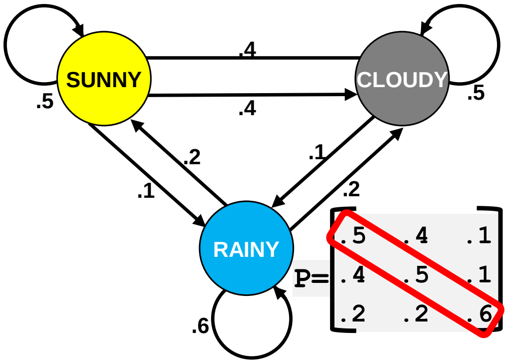

sound waves
phonemes
ah ee s p g ee t oo m ay f t k ih ih ay
words
eye speak tomb I peaked to my eyes pig too mike ice pick two
sentences
I speak to my …



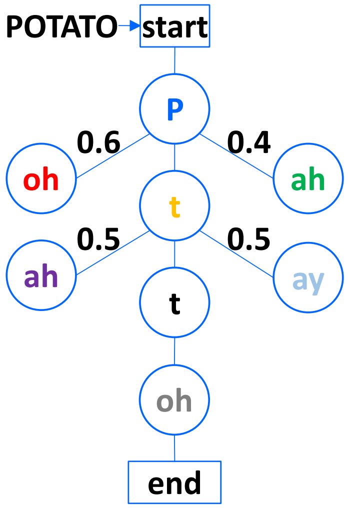
While phonemes look simple, determining them is anything but. Our words are recorded as wavelengths and broken down into digital language signatures called phonemes every few milliseconds. To the left is a wave pattern for the phrase "**How to wreck a nice beach**" broken into phonemes. It happens to also be the same pattern for the phrase "**How to recognize speech**".[71] In other words, the algorithm produces at least two interpretations of the same sound wave and different sets of phonemes.

Phrase	Phonemes
"wreck a nice beach"	r eh k ay n ay s b iy ch
"recognize speech"	r eh k ao g n ay z s p iy ch

Error reduction, such as discerning homonyms (example: "be" and "bee"), is achieved through mathematical probability and statistical analysis disciplines of Hidden Markov Models (HMM) and other techniques to determine the current state based on the output. They help convert sound waves into phonemes and eventually sentences. HMM, while complex and beyond the scope of this paper, use a simpler concept called a Markov chain.



Markov chains, named for Andrey Markov and his early 20th-century work, predicts the future of the process based on its present state. For example, in the diagram to the left, there are three weather states depicted – **sunny**, **cloudy**, and **rainy**. They are given values as one state transitions to another state based on weather observations for a locale and time of year. This Markov chain says if it is **sunny**, there is a .5 (50%) chance that tomorrow will also be **sunny**. Likewise, there is a 10% chance it could **rain** tomorrow and a 40% chance it could be **cloudy**. The percentages always add to 1 in a Markov chain. The **transition matrix P** summaries the probabilities with entries ***Pij*** showing the state transition from ***i*** to ***j***. The **red diagonal** shows the probabilities of getting the same weather on successive days.



In this Markov chain, the word “potato” based on the phoneme probability states gives you multiple renditions. The highest probability is a tie between “**P oh t ah** t **oh**” and “**P oh t ay** t **oh**,” followed by “**P ah t ah t oh**” and “**P ah t ay t oh**.” From a pronunciation standpoint, you might have said "po-TAY-toh,” "po-TAH-toh" or even "pu-TAY-to." If the algorithm “understands” what you are saying, it will process your request correctly, or it could lead you far afield.[72] In the end, a score for the strongest word match from the sound waves is produced. This process is constantly refined through speech training as the number of possibilities based on the vocabulary is large.

In our Markov weather model, the complete set of states are observed. In the potato model, the probability of one phoneme following another is deduced. This is useful since the model takes into account directly observed sound waves and those indirectly hidden in the sound spectrum. Speech recognition has unobservable states that require further sound analysis to determine what was said.

From a word standpoint, you have likely experienced probabilistic typing on your smartphone. On an iPhone, QuickType “predictive text” magically knows what your next word might be.[73] Start a text message with “Happy” and it may offer “Birthday” as your next word choice. Many systems learn from processed data and adjust their Markov probability chain accordingly.

Amazon created a developer's toolset, called the "Alexa Skills Kit" (ASK), which includes the ability to incorporate Alexa's ASR magic into "standard" programs. For example, Domino's pizza used ASK to voice-enable their ordering wizard program allowing its customers to place orders by saying the predefined phrase "open Domino's" in Alexa's presence. ASK currently has over 25,000 skills in its built-in and third-party developer defined library, which includes Starbucks, Uber, Domino's, Capital One and thousands more.[74] Amazon updates their APIs, adds new features, and new skills, making this UI smarter as time goes on.[75] The developer does not need to know how ASR works, just the APIs and how to bind them to their programs. The AVS APIs on the left provide a set of directives and events that map to user activity.[76]

Interface	Interface Description
SpeechRecognizer	The AVS core with a user utterance leveraging the Recognize event.
SpeechSynthesizer	Produces Alexa speech.
Alerts	Set, stop, and delete timers and alarms.
AudioPlayer	Manage and controll audio playback.
Notifications	Display notifications indicators.
PlaybackController	Navigate a playback queue via button affordances.
Speaker	Volume control of a device or application, such as mute.
Settings	Managing Alexa settings on your product, such as locale.
System	Provide client information to Alexa.
TemplateRuntime	Rendering visual metadata.

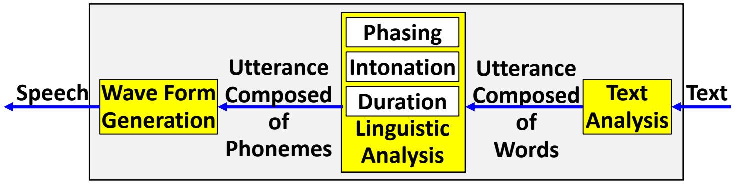
So next time you feel hungry and want to order a Domino's pizza using Alexa, you'll have more of an appreciation of what is going on behind the scenes.

```
Customer: Alexa, open Domino's.
Alexa:    Welcome to Dominos. Would you like to place your Easy Order,
          reorder your most recent order, start a new order, or track your
          order?
Customer: Place my recent order.
Alexa:    It looks like your most recent order is a delivery to [your
          address] and contains two products for nineteen dollars and
          twenty-four cents with a payment method of cash. If you want to
          hear the details of your order, say 'order details' or if you
          would like to place your order, say 'place my order'.
Customer: Place my order.
Alexa:    Your order has been placed. It will be delivered in approximately
          28 to 38 minutes.
```

A few minutes later, you might ask the Alexa VUI the following:

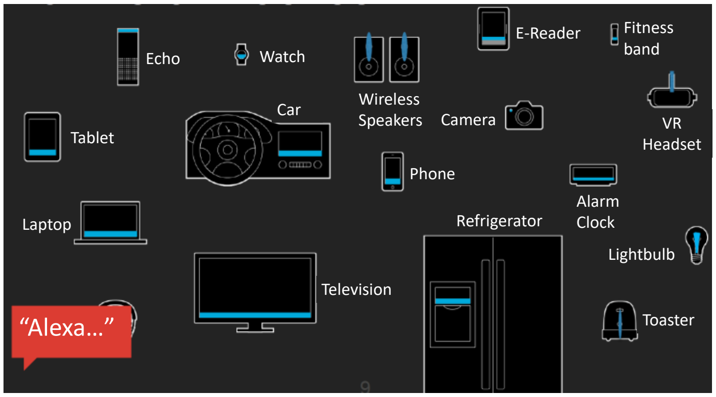
```
Customer: Alexa, open Domino's.
Alexa:    Welcome to Dominos. Would you like to place your Easy Order,
          place your recent order, or track your order?
Customer: Track my order.
Alexa:    Your order is in the oven.
```

In the other direction, we have TTS synthesis.[77] These range from the barely unacceptable to elegant. For example, on the low end of the spectrum, your telephone answering machine concatenates phrases around message counts such as "You have," *digit recording*, "new messages."



As we touched on earlier, Alexa uses Polly to convert text into speech. As a result, you can use Polly's API to announce your name:

```
$ aws polly synthesize-speech --output-format mp3 --voice-id Bruce  --text
"Hello my name is Bruce."   ann.mp3
```



As a user interface, Alexa's open API allows it to front-end a wide array of devices. Their Smart Home Skill API, part of ASK, interfaces Alexa's VUI to turn on lights, so a command like "Alexa, turn on the den light" no longer requires the 1985 double-clap sound-activated "clap on, clap off" invention called "The Clapper."

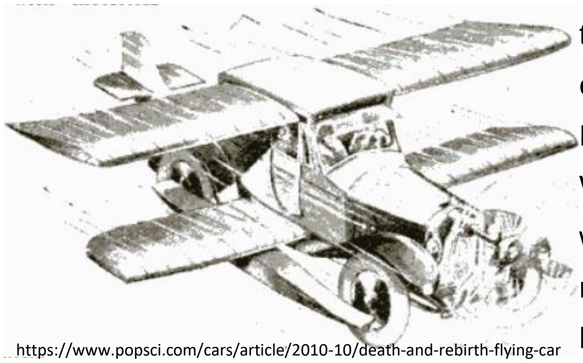
Recently, a "routines" macro capability introduced the concept of bunching together common activities that can be started simultaneously. For example, say "Alexa, good night" and using various skills, Alexa will turn off your television, turn off your lights, and lock your doors.[78]

Automatic Speech Recognition and the Law

While not directly part of the voice user interface discussion, there is an important legal issue that can arise when trying to enhance the man-machine interface, especially with digital assistants like Alexa and others. In November 2015, Victor Collins was found dead in the home of his friend, James Bates, who claimed Collins accidentally drowned in his hot tub. Police served Amazon with a search warrant for the possible witness testimony of Bates's Amazon Echo. As designed, the Echo's ASR streams everything it hears to the AWS cloud once it hears the wake word, but it also records a few words just prior to it detecting the "Alexa" phrase.[79] Amazon claimed that anything the Echo recorded is covered under Bates's protected speech rights under the U.S. Constitution's First Amendment. Bates has consented to the disclosure, and Amazon has released the data to the courts.[80] The case against Mr. Bates was recently closed due to insufficient evidence.[81]

It is worth noting that "Alexa" prefaced conversations can be deleted, and this includes anyone in your house who uses the keyword. You can even review what your teenage son or daughter asks Alexa to do. Amazon states that once deleted, it is removed from AWS servers.[82]

Zero User Interfaces



https://www.popsci.com/cars/article/2010-10/death-and-rebirth-flying-car

As we have seen, most user interfaces are directly linked to the underlying computer technology. While it is difficult to predict the future of technology, as demonstrated by this July 1924 Popular Science illustration of a flying car that World War I flying ace Eddie Rickenbacker thought we would be using by 1944, many believe the next major UI will break the screen dependency we all have, and others go as far as believing that artificial intelligence (AI) will play a critical role . If that comes to be, the world of the user interface is about to undergo some serious changes.

For decades, our computers have had keyboards and monitors. More recently, we have used tablets and smartphones, each with their own types of keyboards and built-in display. We have grown up with embedded computers in thermostats, microwave ovens, calculators, DVRs, and more, and while they did not have traditional input devices nor display screens, they had specific input switches and rigid output panels. Now think about emerging yourself in a world of computing without the input or output devices.

The next step in natural interface computing is beginning and it lessens the importance of display-oriented text and the graphical user interface, and instead focuses on our thoughts, glances, behaviors, voices, and gestures. This new phase also represents a major shift, signifying that humans would no longer try to communicate with a machine in their language but machines would interact with us as people. With the rise of chatbots, voice, and gesture device interactions pioneered in popular gaming systems from Microsoft and Nintendo, we see the beginning of a new movement called the “Zero User Interface” (ZUI).

While a Zero UI sounds like we are doing away with interfaces, the reality is they will always exist. The traditional UI will shift to one of anti-screen/anti-keyboard and be achieved in a more natural way, as though two people were talking. Devices will listen to us, anticipate our wishes, detect our expressions and hand movements, all without a keyboard, mouse, or touchscreen.

The notion of screenless can’t come too soon for some. A large segment of the population is addicted to mobile devices and social media, and that means touches, keyboards, screens, mice, and vibrating phones.[83] We spend 43-53 hours a month juggling 42 smartphone apps and 5 years of our lives on Facebook, SnapChat, and other social media.[84,85] Teenagers check their phones 150 times a day, and users touch their phone 2,617 times daily – a definition of addiction.[86] We sit near people at a restaurant, on a train, or see them driving while typing on and reading from their smartphone. When people wake up, they check their phone for current events, the weather, traffic, and tweets. User dependency on computers of all sizes and shapes has ruined relationships, giving rise to a zombie-class of citizens that do not partake in conversations without glancing at their screens. As we will see, one of the goals of the ZUI is to improve the human-computer interaction, so we can change how we communicate and lessen some of the addiction.

Some believe that in a few years, the notion of downloading apps will be remembered as the "good old days." Technology researcher Gartner, Inc. believes in three years, almost $^1/_3$ of all web browsing will not use a display, and "the average person will have more conversations with bots than with their spouse."[87] As a result, these new always-on UIs will need to be anticipatory, predictive, and context-aware. For example, the Waze automobile navigation application can sync with your calendar to create a "Planned Drive" optimized departure schedule advising you on the best time to leave for an appointment, in addition to its advisory pushed to you based on shifting traffic patterns.[88] Another example could leverage smart tags embedded in your clothing to advise you what to wear and not wear that day based on your day's schedule.[89]

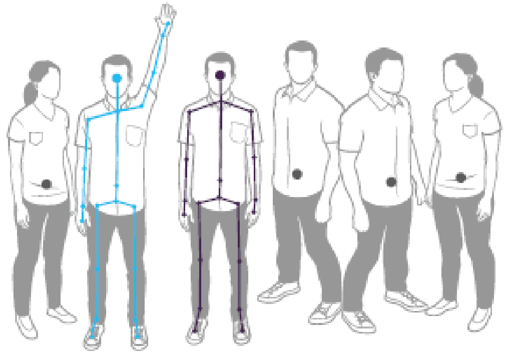
To illustrate the context awareness of Apple AirPods wireless earphones, remove one from your ear and the music pauses or resumes when you put it in, or plays music through the speaker if you are not wearing them – all without touching an app.[90] AirPods use an embedded computer chip programmed for context-aware behavior. Combine Zero UI, TTS, and the concept of AirPods, and you get a new class of product such as Google Pixel Buds and Mymanu's Clik wireless earbuds that leverage the smartphone to provide real-time automatic language translation with the converted speech sent to your ear.[91,92] Imagine being at a conference in Italy and everyone is from a different country. You don't speak Italian, but you can understand what the speaker says and everyone is conversing in their native language. Clik even provides a transcript of the conversation.

Google's CEO Sundar Pichai believes the next UI does not require a keyboard, mouse, or even your voice – it will be an Artificial Intelligence Zero UI.[93] While not gone, the touchscreen world of clicking, typing, and tapping will be augmented by computer vision, voice control, gestures, AI, and haptics (from the Greek *haptikós* – "able to grasp or perceive," the study or use of tactile sensations and the sense of touch as a method of interacting with computers and electronic devices).[94]

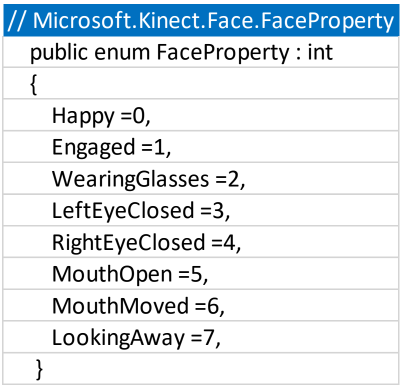
The goal of a Zero UI is simple:[95]

- Remove as much as possible from view so users can live their lives
- Lessen the dependency on computers while still achieving their benefits
- Move the UI closer to our presence, actions, kinesics, and language

Some of these traits are already common in Kinect, Alexa, chatbots, and more. For example,



Microsoft's Kinect can track the movements of 6 people. By using its SDK, you can build Windows 10 apps that leverage a gesture UI based on color detection, infrared, object depth, and facial detection.[96,97] Some of the facial gestures you could incorporate are shown to the right.[98]



```
// Microsoft.Kinect.Face.FaceProperty
 public enum FaceProperty : int
 {
    Happy =0,
    Engaged =1,
    WearingGlasses =2,
    LeftEyeClosed =3,
    RightEyeClosed =4,
    MouthOpen =5,
    MouthMoved =6,
    LookingAway =7,
  }
```

Science fiction fans could imagine recreating the scene from the Tom Cruise movie "Minority Report" where his character uses hand and arm gestures to manipulate his police screen. Other ZUI traits are becoming more commonplace such as vibrating steering wheels when your car deviates from its lane.[99]

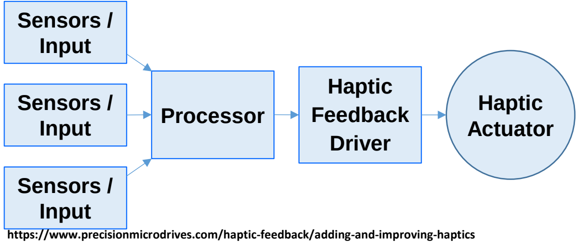
Zero User Interfaces can be a goal, but they may not be a panacea given the differences in computer applications and their design objectives – i.e., a digital watch will likely always have a different UI than a microwave oven. ZUIs also represent the next phase of technologies such as voice control. Right now, Alexa does not know if you are speaking or it's your 5-year old, nor how to string information together. If you ask Alexa to order pizza, it will open a skill that orders pizza, but it is up to the underlying app to figure out who you are. Alexa also lacks the ability to handle multiple questions such as "Alexa, what is the weather and what should I wear?"

Zero UI is already in your everyday world. For example, Samsung's "F" series televisions can be controlled with gestures, facial and voice recognition.[100] Not to eliminate the need for the remote control UI, it augments your remote with natural language voice commands such as "Hi TV, ESPN," "Search for a comedy on CBS," or "I want to watch a Tom Hanks movie".[101] Camera gesture control allows you to interact with your TV

beyond what a simple cursor/mouse UI offers. You can enlarge a picture by separating your hands, change channels and adjust volume with a hand-grasp motion, and navigate menus all with your hand in a “Minority Report” style of interaction.[102] Through facial recognition, it can map age-specific content to each of your children, or it knows you like spy movies, so when you ask for a movie, it might show you a James Bond flick. Their system is also extensible through an SDK that includes sample code, APIs for their web interface, microphone control, and more.[103]

GRAB GESTURE
Clench your hands as if you're holding a fist for this gesture.

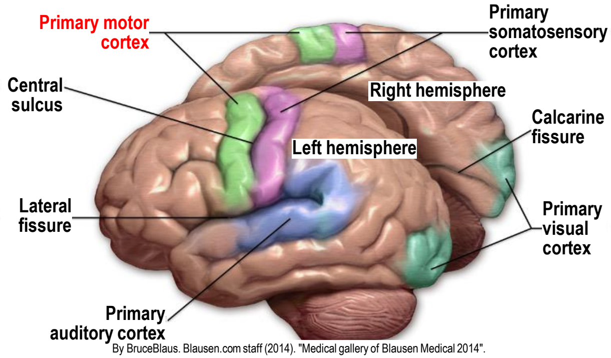
The emergence of Internet-of-Things (IoT), wearables, sensors, data analytics and connected devices will enable new, seemingly magical sets of services. For example, in the future, you may have vibrational “messages” in vehicles equipped with a haptic GPS-enabled steering wheel that leverages a visual map and audible commands to alert your hands using directional vibration to turn left or right at an intersection.[104] A generic block diagram to the left indicates the flow of information from sensors to actuator.



https://www.precisionmicrodrives.com/haptic-feedback/adding-and-improving-haptics

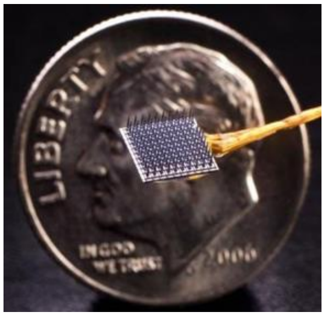
User Interfaces Assist the Disabled

We are watching a miracle take place. People confined to a wheelchair with spinal cord injuries, those with a degenerative disease like amyotrophic lateral sclerosis (ALS, aka Lou Gehrig's disease), and others with debilitating motor disabilities are beginning to utilize a UI that taps into their brain’s electrical activity. Blind individuals are benefiting from sensory augmentation through a haptic UI.

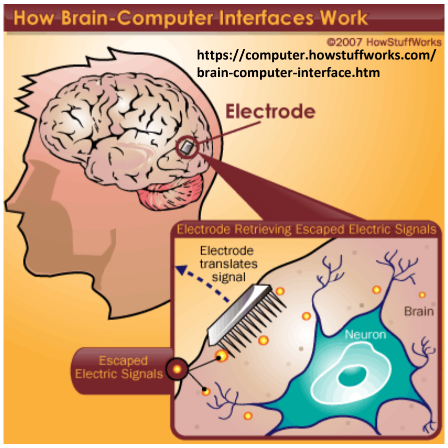


By BruceBlaus. Blausen.com staff (2014). "Medical gallery of Blausen Medical 2014". WikiJournal of Medicine 1 (2). DOI:10.15347/wjm/2014.010. ISSN 2002-4436. - Own work, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=31574257

Brain researchers have found the **primary motor cortex** area, even in severely handicapped individuals, has active neurons capable of generating voluntary arm movements. Shown to the left, the neurons in the **motor cortex** are monitored through a surgically implanted tiny 4-millimeter electrical sensor with 100 tiny electrodes, shown to the right, sitting on the surface of the brain.[105]

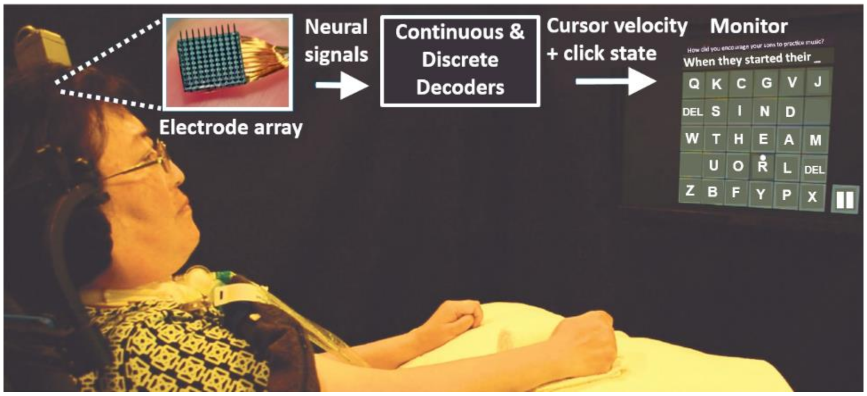


According to researchers at BrainGate, when we move or think about a movement, our brain's motor cortex produces tiny electric impulses that sensors can pick up.[106] Using an MRI scan of the user's brain, an ideal location is found for the implant. A small hole is drilled into the skull and the sensor is implanted as shown to the left. With 100 billion neurons in the brain, connecting the sensor to the right neurons is no easy feat.[107] This chip UI translates distinctive arrangement of impulses with the help of software into mouse-like computer commands with a high rate of success, allowing the user to interact with the world. The user interface allows a user to turn on or off lights, send and receive email, control a TV, move and manipulate a robotic arm, and more.

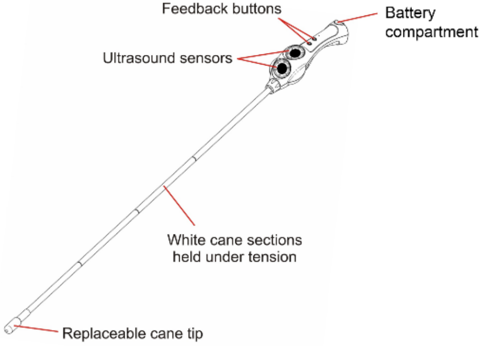

How Brain-Computer Interfaces Work
©2007 HowStuffWorks
https://computer.howstuffworks.com/brain-computer-interface.htm
Electrode
Electrode Retrieving Escaped Electric Signals
Electrode translates signal
Brain
Neuron
Escaped Electric Signals

One of BrainGate's early experiments had a user with a spinal cord injury move the game paddle in the old game of Pong.[108] With the blue paddle moving left and right, as though they were using a joystick gaming interface or the arrows on a keyboard, the user kept the ball bouncing, communicating through the chip UI using just brain signals.

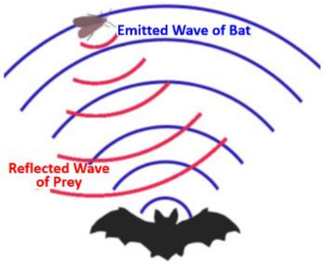
In another test, a 51-year old with ALS typed 24 characters per minute using signals captured from her motor cortex using a Braingate 2 chip implant.[109] An algorithm transforms neural activity into movement intentions, in essence, turning brainwave UI instructions into point-and-click commands.



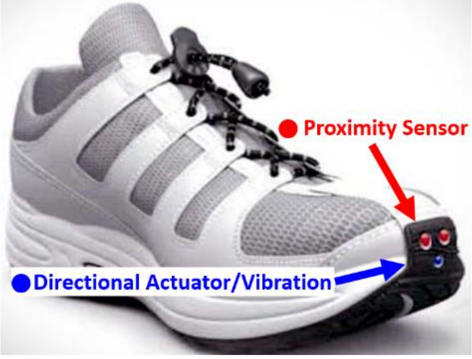
Stanford University researchers using the same chip allowed another paralyzed user to think about the letters in words they wanted to spell and display them on a Google Nexus 9 tablet over a Bluetooth interface.[110] The user could also use their brain signals to move a cursor wherever they wanted on the screen simulating a "touch." They now leverage auto-correct features of the tablet to communicate more easily, interact with apps, and download new apps from the app-store.



www.ultracane.com/download/UltraCane User Guide V1.6 Jan2012-2.html

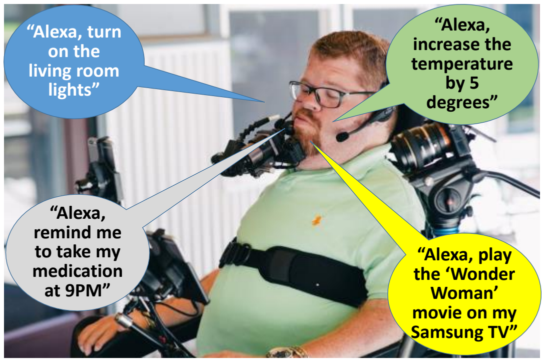


For the blind, the 1921 invention of the white cane traveling aid represented a significant milestone in helping people achieve independence.[111] Almost one hundred years later, there are new product breakthroughs in helping the visually impaired augment their sense of touch. A user's haptic cane employs ultrasonic or infrared waves to directionally vibrate its handle. Similar to how a bat flies in a pitch-black cave and builds a detailed picture of its surroundings, sound waves sent from the cane are reflected back when they find an obstacle.[112] The haptic UI supplies vibration feedback through its handle to give more information about a path or room. Depending on the arrangement of vibrating buttons in the handle, the user knows to veer left or right, and the strength of the pulse tells them how far away the obstacle is. The UI is sensitive, allowing even a blind mountain bike rider on a specially equipped bike to haptically "see" a winding rut covered tree-lined track at a moderate speed without incident.[113]



Haptic concepts can be built into items such as wristbands, glasses and even shoes. Bluetooth sensors in haptic shoes use smartphone processing to guide a visually impaired wearer down the sidewalk, up and down stairs, around and across street corners, with GPS accuracy, and without a cane.[114] Made by Lechal, the wearer speaks their destination into an app, and vibrations in the left and right shoe tell them where to turn, or warn of an obstacle in their path.[115]

Other devices are assisting the blind through color, brightness, and temperature sensors built into a feedback cane. A user can pick out their color-coordinated clothing for the day by moving the cane near their wardrobe and pressing the color button to find out if a dress is blue, or a shirt is pink. It can also determine the brightness of your surroundings from a safety perspective, and if their beverage is hot.[116]

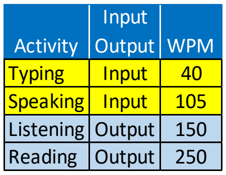

"Alexa, turn on the living room lights"
"Alexa, increase the temperature by 5 degrees"
"Alexa, remind me to take my medication at 9PM"
"Alexa, play the 'Wonder Woman' movie on my Samsung TV"

Even simple devices like Alexa turn out to be useful to quadriplegics, visually impaired, users with limited mobility or dexterity, and other disabilities. They can give commands to their thermostat, door locks, entertainment devices and more. Users can ask "Alexa, give me some news" or "Alexa, please wake me at 8 AM".[117]

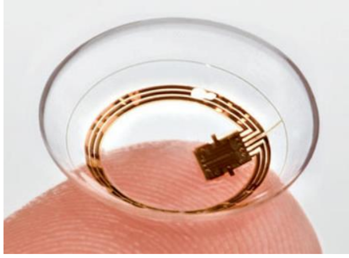
Future

While we may not abandon user interfaces tied to a mouse and keyboard, we are experiencing a UI journey, similar to the one's computers and the internet has taken. Modern UIs focus on psychology and technology and are conduits for transactional apps using APIs that help us order food or hail a ride. "Punch card" veterans transitioned to text "green screen" interfaces, graphical desktops and the internet. Application installation and "pulling" website content is shifting to devices that "push" content towards us.[118] The era of UI transformation from a GUI and mouse to one that parallels real life has begun. Humans converse through comments, questions, and other formal and informal modes of communication, and we are experiencing natural language capabilities that allow us to have conversations with our devices.

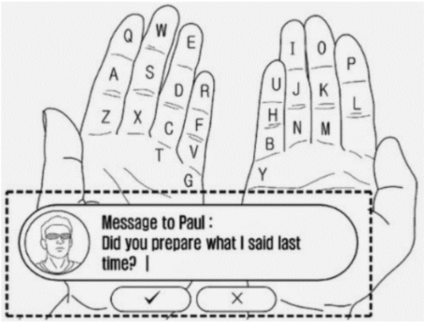
Transformation to seamless interactive technology is inevitable and desirable since we generally type 40 words per minute (WPM), speak 105 WPM, listen 150 WPM, and read 250 WPM, which means less typing and more speaking increases productivity.[119] Smartphones are a step in the right direction, yet they still rely on a finger GUI, and when they need our attention, they can pose a risk such as texting while driving. VUIs like Siri are in their infancy, yet they get us closer to the free-flowing conversation change we want and a faster "WPM" than typing. It's great when Alexa tells us an answer, but faced by a complex query, it uses a multimodal approach with a reply displayed on a screen.[120] Needless to say, a VUI can be inappropriate in public places, noisy venues, meetings, or when privacy is a concern. Here are efforts that transform how technology will accommodate us.



Activity	Input Output	WPM
Typing	Input	40
Speaking	Input	105
Listening	Output	150
Reading	Output	250

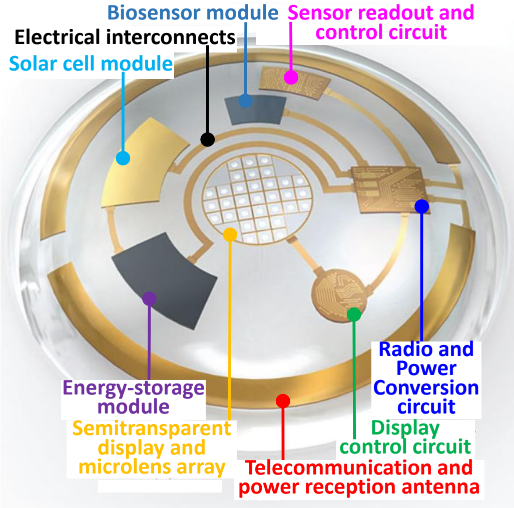


Companies are working on digital contact lenses. Just as the smartphone evolved from a flip phone to a gadget with web access, turn-by-turn navigation, video conferencing and more, the contact lens UI may move beyond vision correction. Samsung's "Gear Blink" smart lenses have a display, camera, antenna, sensors to detect movement, and project augmented reality right to the user. Their lens leverages smartphone processing power and could conceivably overlay driving directions on the road similar to football's TV first-down yellow line markers. It would allow us to use fingers as a virtual keyboard to type in thin air or take a photo by blinking.[121,122,123]

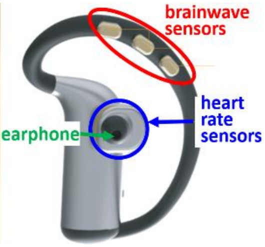


Message to Paul :
Did you prepare what I said last time? |

Let's imagine this digital contact lens has a **biosensor** that collects user temperature, blood pressure, pulse, and respiration rates. It currently senses the brain's desire for a cheese pizza.[124] **LEDs**, normally transparent for vision correction, now display pizza sizes using the **display circuits** and smartphone data. The brain picks a large pie and has the **telecommunication and power reception antenna** open an internet connection using APIs from the **radio and power conversion circuit**. With the order placed, the **antenna** collects transmitted energy and **solar power** charges the **capacitor** to power the lens.

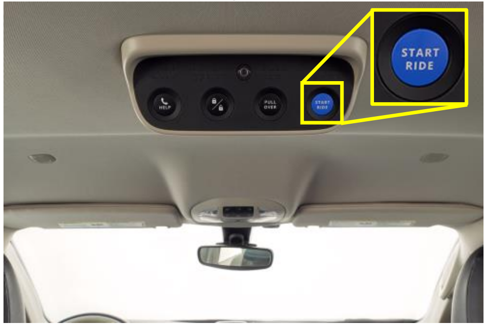

Biosensor module
Sensor readout and control circuit
Electrical interconnects
Solar cell module
Radio and Power Conversion circuit
Energy-storage module
Semitransparent display and microlens array
Display control circuit
Telecommunication and power reception antenna

Someday, we all may wear a noninvasive Bluetooth brain-computer UI (BCI) – a direct neural interface worn around your ear that acts like a brain USB port, allowing you to plug in a library of information. Similar to the digital contact lens, it picks up our thoughts and automatically uses our phone for computing power. It might look something like the images to the left.[125] Thinking about making a special BBQ sauce for ribs, or how to repair a leaky faucet? Your smartphone makes the connection with a digital chef or gets you in touch with a virtual plumber, all through your bidirectional BCI. Hungry for a burger while you are driving? A thought directs you to a drive-in restaurant along your route. BCIs will require customization since our brains are unique.



Since the first internal combustion engine equipped car hit the road 130 years ago, a human has been behind the wheel.[126] Now with autonomous cars seemingly right around the corner, their UI will be key to its success since most will not have a steering wheel or brake pedals. In Arnold Schwarzenegger's "Total Recall" movie, his character Quaid hops into a taxi and is greeted by "Hello, I'm Johnny

Cab. Where can I take you tonight?"[127] The robot driver's Conversational User Interface (CUI), a highly advanced Automatic Speech Recognition system designed solely for audio responses chatbot-style (think of the ELIZA example), requests Quaid's destination. Drawing parallels to Johnny Cab, Waymo (a subsidiary of Google's parent company Alphabet) has been working on a self-driving car for years.[128] In 2017, riders got into the second row of a Chrysler minivan and the only UI they saw were four buttons - a **blue** one labeled "**START RIDE**", and others for "pull over," "lock/unlock," and "help." They had a display at their seats welcoming them to this autonomous vehicle. The rider entered their destination, perhaps through a CUI Google app, pressed the blue button, and off they went.



This type of future innovation holds great promise for society. Imagine a blind man entering the car with no steering wheel and no pedals.[129] They simply tell the car where they want to go, and it navigates everyday traffic and arrives at the destination. The passenger's freedom of mobility is restored without the need for a sighted driver. The same freedom is extended to children, those who need a designated driver, and those without a driver's license. The UI masks the underlying complexities of sensors (radar, camera, LiDAR, and microphones) that scan for other vehicles, people, and obstacles. The car plots a GPS route using an AI algorithm to direct the vehicle around complex urban roads and communicates with other autonomous and non-autonomous cars along the way. In many ways, the driverless car is like the development of the elevator as it transitioned from requiring an elevator operator to a self-service UI – the interface takes care of the underlying complexity. As demonstrated by manufacturers shaping the future of a driverless car, AI, inter-vehicle communications, and a Zero UI will play key roles in ensuring autonomous cars are safe.

User interfaces continue to progress. Just like the car, which replaced the tiller with a steering wheel, computer UIs have become more powerful. For instance, a key fob unlocks our car door, and some of us press the "engine start" button instead of using a physical key. Someday, car doors will swing open when it senses we need to get into the car, perhaps through a zero user interface.

Gesture control UIs, such as those we reviewed with TVs, might someday allow us to rotate screen images for a more personal view of a car or jewelry. Imagine using a finger to circle a TV advertisement for a couch, expand the image by separating our hands, rotate it with a circular hand motion, change its color, and place it in a 3D virtual view of our living room.

In the distant future, a BCI might allow quadriplegics to be reunited with their own muscles, or assist patients in controlling their Parkinson’s and epilepsy symptoms, restoring their lives to some semblance of what it used to be.

Even further out in time, humans may perfect the artificial intelligence UI to the point where it can predict what we want to accomplish before the thought pops into our heads.

Conclusion

This paper has touched upon some of the changes in how humans interact with all sorts of computers. We have seen how desktop environments favor certain user interfaces, handheld devices require hand-oriented interfaces, digital assistants favor verbal interfaces, and those with disabilities have their own special needs where sensors play a major role.

In our world, our computing needs are always changing because, at the heart of human-computer interaction, humans are changing. The status quo is being altered by advances in technology, economics, politics, preferences, education, environment, and dozens of other factors. Human needs will continue to dictate the pace of change of user interfaces.

We are witnessing an evolution in computing where instead of humans adapting to the needs of computers, the computers are adapting to us. Just as computers were built to communicate with us through printouts, keyboards, screens and touch, they will likely be aided by new Zero User Interfaces to enable them to further adapt by understanding our thoughts and emotions. These incredible technological transformations in our daily lives will only continue to occur in the years to come.

Footnotes

---

[1] https://www.techspot.com/guides/477-xerox-parc-tech-contributions/
[2] https://en.wikipedia.org/wiki/Hackamore
[3] https://www.bitlessbridle.com/Evolution_of_bitless_equitation.pdf
[4] https://www.equisearch.com/articles/double_bridle_071708
[5] https://en.wikipedia.org/wiki/Saddle
[6] https://www.thehenryford.org/collections-and-research/digital-collections/artifact/252049/
[7] http://www.history.com/this-day-in-history/henry-ford-test-drives-his-quadricycle
[8] http://www.otisworldwide.com/pdf/aboutelevators.pdf
[9] https://www.investopedia.com/terms/m/mooreslaw.asp
[10] https://en.wikipedia.org/wiki/Jacquard_loom
[11] http://whatis.techtarget.com/reference/History-of-the-punch-card
[12] http://www.quadibloc.com/comp/cardint.htm
[13] http://blog.usabilla.com/short-history-computer-user-interface-design/
[14] https://en.wikipedia.org/wiki/Keypunch
[15] https://en.wikipedia.org/wiki/Keypunch
[16] http://www.computinghistory.org.uk/det/613/the-history-of-the-computer-mouse/
[17] http://www.explainthatstuff.com/computermouse.html
[18] Based on http://www.qb64.net/wiki/index.php?title=Mouse_and_Keyboard_Input
[19] https://www.qb64.org/articles/articlemouse.txt
[20] http://www.tandfonline.com/doi/abs/10.1080/00140136708930868
[21] https://en.wikipedia.org/wiki/HP-150
[22] http://whatis.techtarget.com/definition/natural-user-interface-NUI
[23] https://www.interaction-design.org/literature/article/natural-user-interfaces-what-are-they-and-how-do-you-design-user-interfaces-that-feel-natural
[24] //www.electronicproducts.com/Sensors_and_Transducers/Sensors_and_Transducers/Resistive_VS_Capacitive_Touchscreens.aspx
[25] https://developer.apple.com/documentation/uikit/uiresponder/1621142-touchesbegan
[26] https://www.iso.org/search/x/query/9241
[27] https://uxplanet.org/z-shaped-pattern-for-reading-web-content-ce1135f92f1c
[28] http://www.nonlinearcreations.com/Digital/how-we-think/articles/2014/07/What-I-learned-about-bad-interfaces.aspx
[29] http://xenon.stanford.edu/~lswartz/vincennes.pdf
[30] http://www.dodccrp.org/events/10th_ICCRTS/CD/papers/289.pdf
[31] "User Interface Design and Evaluation" Debbie Stone, Caroline Jarrett, Mark Woodroffe, Shailey Minocha. ISBN 0120884364, p.7
[32] https://www.bulldogreporter.com/over-a-third-of-americans-confess-to-verbal-or-physical-abuse-of-their-computers-ne/
[33] http://technewstube.com/pc-world/586553/a-tribute-to-windows-8-if-it-hadnt-been-so-bad-windows-10-wouldnt-be-so-good/
[34] http://www.vcalc.net/touchtone_hf.pdf
[35] http://www.todayifoundout.com/index.php/2015/10/keypad-numbering-different-phones/
[36] http://retrocalculators.com/sundstrand.htm
[37] http://www.usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces
[38] https://www.slideshare.net/naveensagayaselvaraj/user-interface-design-66894631
[39] http://www.wseas.us/e-library/conferences/2005argentina/papers/503-115.doc
[40] http://beijerinc.com/pdf/whitepaper/interface_design_best_practices.pdf
[41] http://info.localytics.com/blog/app-user-retention-improves-in-the-us
[42] http://www.historyworld.net/wrldhis/PlainTextHistories.asp?historyid=ab13
[43] http://www.chakoteya.net/movies/movie4.html
[44] https://blog.infostretch.com/view-from-the-labs-voice-user-interfaces-a-short-history-and-a-bright-future/
[45] http://ieeexplore.ieee.org/document/1369165/
[46] https://news.harvard.edu/gazette/story/2012/09/alan-turing-at-100/
[47] http://www.masswerk.at/elizabot/
[48] https://www.pcworld.com/article/243060/speech_recognition_through_the_decades_how_we_ended_up_with_siri.html
[49] http://www.dragon-medical-transcription.com/history_speech_recognition.html
[50] https://en.wikipedia.org/wiki/Dragon_NaturallySpeaking
[51] https://developer.amazon.com/alexa-skills-kit
[52] https://developer.amazon.com/blogs/alexa/post/Tx205N9U1UD338H/introducing-the-alexa-skills-kit-enabling-developers-to-create-entirely-new-voice-driven-capabilities
[53] https://www.technologyreview.com/s/602896/fedex-bets-on-automation-as-it-prepares-to-fend-off-uber-and-amazon/
[54] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/overviews/understanding-custom-skills
[55] https://www.slideshare.net/aniruddha.chakrabarti/amazon-alexa-building-custom-skills
[56] https://aws.amazon.com/blogs/aws/polly-text-to-speech-in-47-voices-and-24-languages/
[57] https://www.slideshare.net/AmazonWebServices/fsi300-bringing-the-brains-behind-alexa-to-financial-services
[58] https://aws.amazon.com/lambda/details/
[59] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/built-in-intent-ref/built-in-intent-library
[60] www.recode.net/2017/5/31/15720118/google-understand-language-speech-equivalent-humans-code-conference-mary-meeker
[61] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/built-in-intent-ref/built-in-intent-library
[62] http://www.chronicle.com/blogs/linguafranca/2013/05/09/natural-language-processing/
[63] https://plato.stanford.edu/entries/turing-test/
[64] http://blogs.aspect.com/dont-confuse-speech-recognition-with-natural-language-understanding/

---

[65] https://www.wired.com/2011/01/birth-of-the-microphone/
[66] http://electronics.howstuffworks.com/gadgets/high-tech-gadgets/speech-recognition.htm
[67] http://www.icc-languages.eu/component/attachments/download/123
[68] http://www.fahs.surrey.ac.uk/languages/spanish-phonetics/advanced-phonetics-7-2/spanish-phonemes.htm
[69] https://www.youtube.com/watch?v=xiqUVnXExTQ
[70] https://www.ethnologue.com/
[71] https://aretoulaki.wordpress.com/2010/05/20/maria-aretoulaki-speech-recognition-for-dummies/amp/
[72] https://stats.stackexchange.com/questions/148023/markov-chains-vs-hmm/148034
[73] http://www.ecormany.com/picomac/2016/10-quicktypes-quirky-collocates
[74] https://developer.amazon.com/alexa-skills-kit
[75] https://developer.amazon.com/alexa-voice-service
[76] https://developer.amazon.com/docs/alexa-voice-service/api-overview.html
[77] https://en.wikipedia.org/wiki/Speech_synthesis
[78] https://techcrunch.com/2017/09/28/alexas-routines-will-combine-smart-home-control-with-other-actions-like-delivering-your-news-and-weather/
[79] https://www.amazon.com/gp/help/customer/display.html?nodeId=201602230
[80] http://www.smithrolfes.com/Uploads/files/Alexa Please Help Solve My Case; CBA Report; September 2017 .pdf
[81] http://www.cnn.com/2017/11/30/us/amazon-echo-arkansas-murder-case-dismissed/index.html
[82] https://www.amazon.com/gp/help/customer/display.html?nodeId=201602040
[83] http://www.economist.com/node/21549904
[84] http://www.nielsen.com/us/en/insights/news/2014/tech-or-treat-consumers-are-sweet-on-mobile-apps.html
[85] http://mediakix.com/2016/12/how-much-time-is-spent-on-social-media-lifetime/#gs.5QxT17M
[86] http://www.businessinsider.com/dscout-research-people-touch-cell-phones-2617-times-a-day-2016-7
[87] https://www.gartner.com/smarterwithgartner/gartner-predicts-a-virtual-world-of-exponential-change/
[88] https://techcrunch.com/2016/03/16/waze-now-tells-you-when-to-leave-thanks-to-its-new-planned-drives-feature/
[89] http://abcnews.go.com/WNT/story?id=131144&page=1
[90] https://www.apple.com/airpods/
[91] https://www.cnbc.com/2017/10/04/google-translation-earbuds-google-pixel-buds-launched.html
[92] https://www.youtube.com/watch?v=vEQK_urrGZs&app=desktop
[93] https://digiday.com/marketing/wtf-is-zero-ui/
[94] http://www.dictionary.com/browse/haptics
[95] https://careerfoundry.com/en/blog/ui-design/what-is-zero-ui/
[96] http://go.microsoft.com/fwlink/?LinkID=247735
[97] https://channel9.msdn.com/Blogs/k4wdev/Media-Capture-Frames-API-and-Kinect-in-the-Windows-10-Anniversary-Update
[98] https://github.com/HAZARDU5/kinect-unity-test/blob/master/Assets/Standard Assets/Microsoft/Kinect/Face/FaceProperty.cs
[99] https://www.engadget.com/2011/10/29/fords-driver-alert-system-keeps-you-in-the-right-lane-recommen/
[100] http://www.samsung.com/au/support/skp/faq/1039747
[101] https://www.youtube.com/watch?v=wDmosRnEfiw
[102] http://www.samsung.com/ph/smarttv/common/guide_book_3p_si/grab.html#
[103] http://developer.samsung.com/services-and-apis
[104] https://www.youtube.com/watch?v=FKSOO7zXGnM
[105] https://www.youtube.com/watch?v=9oka8hqsOzg&feature=youtu.be
[106] http://researchscript.com/wp-content/uploads/2017/03/IJRCS040131.pdf
[107] https://faculty.washington.edu/chudler/what.html
[108] https://www.youtube.com/watch?v=Nj9iQZBnmU4
[109] https://elifesciences.org/articles/18554
[110] https://www.youtube.com/watch?v=r5u05Q8p3b8
[111] http://www.acb.org/tennessee/white_cane_history.html
[112] https://commons.wikimedia.org/wiki/File:Bat_echolocation.jpg
[113] https://www.youtube.com/watch?v=yC-oJqAIO3M
[114] https://blog.ted.com/video-haptic-feedback-shoes-guide-blind-walkers/
[115] https://trackimo.com/bluetooth-shoes-for-the-blind/
[116] https://www.youtube.com/watch?v=TUIg2KFZozA
[117] https://gettecla.com/blogs/news/how-tecla-e-enhances-amazon-alexa-for-users-with-physical-disabilities
[118] https://chatbotsmagazine.com/the-future-of-ui-bots-conversation-voice-zero-ui-and-the-end-of-the-mobile-apps-defa5dcc09f5
[119] http://www.humanfactors.com/newsletters/human_interaction_speeds.asp
[120] https://www.youtube.com/watch?v=b8pTqhfbo40
[121] https://www.reviewofoptometry.com/article/cls-beyond-vision-correction-connecting-to-the-internet-of-things
[122] https://www.sammobile.com/2014/09/10/tizen-powered-gear-blink-rumored-to-launch-in-march-2015/
[123] http://technave.com/gadget/Rumours-A-Samsung-smart-contact-lens-in-the-making-6859.html
[124] https://www.slideshare.net/geets13/empowered-lens
[125] https://www.indiegogo.com/projects/brainno-smart-wearable-for-brain-training#/
[126] https://en.wikipedia.org/wiki/History_of_the_automobile
[127] https://www.youtube.com/watch?v=xGi6j2VrL0o
[128] https://en.wikipedia.org/wiki/Waymo
[129] http://www.dailymail.co.uk/sciencetech/article-4029838/Get-ready-Waymo-self-driving-cab-Google-creates-new-firm-autonomous-car-technology.html

Dell EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” DELL EMC MAKES NO RESPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying and distribution of any Dell EMC software described in this publication requires an applicable software license.

Dell, EMC and other trademarks are trademarks of Dell Inc. or its subsidiaries.