AHI Day 4 -- Usability 101 & UI Prototyping

Index

1.0 Usability 101

Usability in games is very much like an iceberg encountered at sea -- what we see is a massive chunk of frozen water jutting from the water, but as you probably know 90% of the iceberg is actually below the surface. Like an iceberg, in games the **interface** is obvious at the surface, but there is much, much more "**performance load**" below the surface.

1.1 Interface is comprised of three aspects, vision, hearing (both perceptual) and manipulation or input (kinematic).

Vision, the HUD (Heads Up Display) or the screen GUI (Graphic User Interface) provides visual data to the user. This is the human primary sense in humans, coupled closely with hearing. Vision is discussed at greater length in section 3, Science of Usability.

Hearing or the soundscape includes the audio components used for prompts and alerts, confirmations and immersion (ambience). This is a primary sense outside of the visual range and also serves as a profound supporting sense within the visual range ("I see a duck and I hear a duck, so it is almost certainly a duck).

Manipulation or control devices come in a vast variety, from keyboard and mouse to any of the numerous console controllers to swipe screens and so on.

1.2 Principles of User Interface Design (Arthur Abraham's "Humane User Interface", 2004)

These are some general principles that help designers focus their efforts. It might seem like commercial product just happen, but they generally abide by these guidelines to make the user comfortable and effective.

- **The Structure Principle**: Design should organize the user interface purposefully, in meaningful and useful ways based on clear, consistent models that are apparent and recognizable to users, putting related things together and separating unrelated things, differentiating dissimilar things

and making similar things resemble one another. The structure principle is concerned with overall user interface architecture.

- **The Simplicity Principle**: The design should make simple, common tasks easy, communicating clearly and simply in the user's own language, and providing good shortcuts that are meaningfully related to longer procedures.
- **The Visibility Principle**: The design should make all needed options and materials for a given task visible without distracting the user with extraneous or redundant information. Good designs don't overwhelm users with alternatives or confuse with unneeded information.
- **The Feedback Principle**: The design should keep users informed of actions or interpretations, changes of state or condition, and errors or exceptions that are relevant and of interest to the user through clear, concise, and unambiguous language familiar to users.
- **The Tolerance Principle**: The design should be flexible and tolerant, reducing the cost of mistakes and misuse by allowing undoing and redoing, while also preventing errors wherever possible by tolerating varied inputs and sequences and by interpreting all reasonable actions.
- **The Reuse Principle**: The design should reuse internal and external components and behaviors, maintaining consistency with purpose rather than merely arbitrary consistency, thus reducing the need for users to rethink and remember.

1.3 Performance Load

This is the amount and ease of processing required for success, both mentally (Cognitive) and physically (Kinematic).

Cognitive load is the amount of mental activity required in accomplishing a goal, including perceptual tasks, memory requirements and problem solving. For example, graphical user interfaces have dramatically decreased the cognitive load involved in performing tasks on computers -- in the early days of computers, users had to learn long lists of commands and then type them into the computer in very specific sequences and structures, whereas now we browse from a menu of commands (helpfully broken into easy to understand hierarchies) or double-click on an icon to use an application. A large part of the success of computers as mass market devices is the reduction in cognitive load.

There are a number of strategies that can be used to reduce mental effort, including minimizing visual noise (and maximizing signal), chunking information that needs to be remembered (as in browser menus), using memory aids (i.e. icons or tool-tips) to assist recall and problem solving, and automating tasks which make heavy demands on computation and memory (.

Kinematic load is the amount of physical activity required to accomplish a goal, and is impacted by the number of steps or movements involved in the task and the amount of force required for each of those steps. When the telegraph was invented, and communication was conducted through a mechanical ‘tapping’ letter by letter, the number of taps required to communicate a message was the kinematic load. Samuel Morse specifically designed Morse Code to minimize this load by keeping the simplest codes for the most common letters and more sophisticated code for uncommon letters (specifically, E is a single dot, whereas Q is much longer dash-dash-dot-dash). This design reduced the physical effort required to send messages, significantly reducing transmission times and error rates.

Other strategies for reducing kinematic load include reducing the number of steps required in a process, minimizing the range of motion and travel distances (Fitts’s Law), and automating tasks which are highly repetitive.

1.4 UI vs. UX

UI (User Interface) and UX (User Experience) are often mistakenly considered interchangeable terms, though they are distinct and separate aspects of the conduit between player and game. UI is the

mechanical devices that convey information and control (the screen HUD, keyboard, mouse, console controller, etc) through which a user interacts with the game. UX is a measure of how intuitive and enjoyable those interactions are made through refinement.

It is easiest to envision the two as vehicles: UI is a car, 4 wheels an engine and a chassis that can be used to travel faster and further than walking. UX is the condition and amenities available to the specific car -- a "beater" will get you there, but not in the style and comfort of a luxury vehicle designed to enhance the experience.

1.5 What is Usability?

The hierarchy of usability goes from the overarching goal (satisfaction) to the finishing touches (efficiency).

Satisfaction – an experience the user will remember and want to repeat!

Error Free – low error rate makes the product great!

Learnability – quick and easy to learn is preferred, recovery essential.

Memorability – learn it once and never have to learn it again, but sometimes be reminded.

Efficiency – maximum productivity with minimum effort – less steps same result.

1.6 Usability Goals

Functionality is the first and most obvious goal in usability, "it should work." Part of this I the mechanical function (code), but a less apparent aspect is the specific crafting of the interface to purpose (a principle known as "affordance).

Simplicity is the streamlining of usage by reducing extraneous steps in a process.

Forgiveness is an essential goal in usability, facilitating the user WHEN (and not if) they make the inevitable mistake. Yes, they made a mistake, but they should not be punished for it despite how early designers feel about the player suffering for their failures. The common convention in user forgiveness is confirmation ("save document before closing") and the "escape hatch" to allow a player to "go back" or deep software to "go to main menu."

Habitation (convention) is a saving grace in usability design, applying the tried and true methods refined from decades of global experience. It is generally a good idea to start with conventions in UI rather than "reinventing the wheel" every time you make a game. The screen HUD and controls of a given genre are more often than not established methods which could be used "as is" or as a launching point for experiments on variation.

The most crucial question to apply to deviation of the norm is "are we improving on the convention or just deviating?" If the answer is only deviation, then you're probably better off just staying with convention.

Perception

Hearing (Auditory Perception, or Audition) is a primary sense with which you are likely very familiar. However, people generally take hearing for granted and don't realize some of the interesting nuances of the sense and its neurological associations. It is a supporting sense to modern people (not as leery of predators hunched in the shrubbery), but actually a profound aid to the primary sense of vision.

Alert and direction finding. Hearing provides a full environmental perception, or a "360 sense." Something that is out of the visual range ("behind you") or within the normal visual range ("in front of you") but obscured can be detected and located (by Sound Localization) with hearing. Very much like we use two eyes to determine distance (Binocular Vision), we use two ears to identify direction.

In games, use sound to alert the player to what they do not see (or might not see) and reinfoce what they do see.

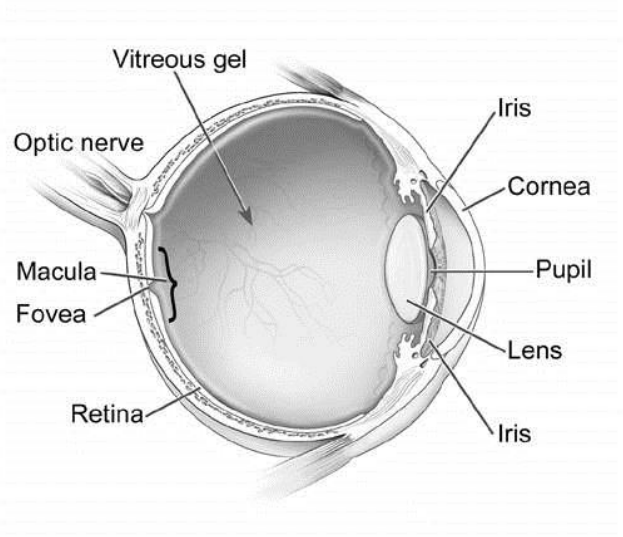
Filtering. The "cocktail party effect" (amid the din of numerous conversations, you can focus on the conversation with one person) is the result of "lateral inhibition" -- brain cells in the primary auditory cortex can both turn down the noise and increase the gain on the auditory signal. This is not unique to hearing as visual and olfactory (smell) inputs are similarly processed.

In games (unlike the real world where you do not have as much control), reduce sounds to effective (for play) and mitigated ambiance (immersion).

Vision -- Foveal vs. Peripheral is a ponderous topic, but distilled here for the sake of expediency. You are certainly encouraged to look deeper into this fascinating subject with a bit of your own research, but the basics are covered here. We are all familiar with vision from our personal usage and with the premise of peripheral vision (seeing out of the "corner of your eye").

To begin we must establish the structures of the retina. Rods and Cones are two of the three receptive structures in the retina (the third is Retinal Ganglion Cells, but that is peripheral to the lesson). Rods are for low light conditions and detect black and white while the Cones are for higher light conditions and detect color (as a mnemonic device C is for Cones and for Color). Rods (low light or black and white) are more receptive to contrast and movement becomes more prominent. Cones detect color and in so doing are receptive to the greater nuance of visual detail (shapes and contour being th most conspicuous aspects). Ultimately we take in this glut of data and our internal filters process it as one lump sum of data without us knowing or caring (much like driving a car doesn't require that you know what a catalytic converter does).

Foveal Vision. You don't hear the term "foveal vision" much outside dusty pedantic conversations, but when you do hear it think "normal vision". The fovea is a divot in the retina at the rear of the eye and just offset from the optic nerve (optic disc). It is the most specific point of visual contact in our (primate) eye. When you "look at something" you are actually using this point of alignment, the fovea through the lens to the point in the universe you are inspecting. Within the fovea are a concentration of Cones and a lesser density of Rods, resulting in less light sensitivity and more detailed visual data.



Peripheral Vision. The vast stretch of retina outside of the fovea handles what we call "peripheral vision." Though we don't distinctly know it, this aspect of vision is less color sensitive and more motion sensitive. Quite simply, it is used to attract our foveal vision.

In games, foveal vision (high definition vision) is roughly the size of your fist on the screen. Of course where you are looking on the screen determines what is inside this range, but anything outside of this area is only being seen with your peripheral vision (motion and less color).

1.7 Scientific Principles

Aesthetic Usability Effect describes a phenomenon in which the user perceives more-aesthetic designs as easier to use than less-aesthetic designs. "A fresh coat of paint helps sell" is a more common way to explain the principle. A prototype with polished graphics, animations, rich sound (effects and music) will often yield less critical feedback.

Fitts's Law is a descriptive model of human movement which predicts that the time required to rapidly move to a target area as a function of the ratio between the distance to the target and the width of the target. It is used to model the act of *pointing*, either by physically touching an object with a hand or finger, or virtually, by pointing to an object on a computer monitor using a mouse or other pointing device.

In short, Fitts's Law states things that are smaller or farther away are more difficult to accurately target. This is reflected in the common adage "can't hit the broad side of a barn", or being incapable of targeting the large target. This applies to game usability in that the developer makes moving the mouse and clicking on a target easier by making the target closer to an origin point (other important targets) and or larger (the size of the cursor itself is a good rule for the smallest a target should be when expecting the user to accurately interact under stressed conditions).

Hick's Law (or the **Hick–Hyman Law**) describes the time it takes for a person to make a decision as a result of the possible choices he or she has: increasing the number of choices will increase the decision time logarithmically. The Hick–Hyman law assesses cognitive information capacity in choice reaction experiments. The amount of time taken to process a certain amount of bits in the Hick–Hyman law is known as the *rate of gain of information*.

In short, Hick's Law states that the time to make a decision is directly impacted by the number of and the clarity (understandability) of the available choices. Fewer and more clear choices aid in the decision making process.

Stroop Effect refers to the reduction in reaction time as a result of informational "noise" or conflicts. In short, the user is slowed by confusing or contradictory input. This directly represents the premise advanced by Hick's Law (above), where the decision process is made more efficient by clear choices and the reduction of distraction.

You might have taken a Stroop test back in middle or high school. The most common presentation of this principle is a word (the name of a color, such as "red"), but the color of the font (text) is a different color (like green). Before the visual is revealed, the subject is asked to identify the color of the text of the word (green in this example). When the visual is revealed the subject speaks their answer and hopefully can recognize the resulting lag in their response -- the lag results from the confusion of the written word which immediately has meaning in the reader's mind conflicting with the answer to the actual question, which is a different color. Google image search "Stroop effect" and see if that rings a bell.

OODA Loop refers to the decision cycle comprised of Observe, Orient, Decide, and Act. It is a principle developed in the military (USAF Colonel John Boyd) for combat but was quickly identified as the process for any competitive engagement (like business cycles as one example). It is also applicable to strategic and tactical consideration in games.

Understanding the stages and nuance of this cycle can help a designer better identify and rectify usability issues in their product. What does the player experience, how is that information frames, what options are available and how are they weighted and what are the results of actions are an encapsulation of the user experience.

Observe takes in all available information, including extraneous and distracting input.

- **Perception** (specifically audible and visual input, though maybe others) is the base input.
- **Noise** is extraneous or distracting input.

Orient apples existing knowledge and experience to the available information. Here conventions (habituation) plays a considerable role improving user reaction.

- **Experience** (context) is the key to success (says the 13th level fighter to the 2nd level ogre).
- **New information** should be afforded the luxury of "safe introduction" and where needed repetition.

Decide includes the formulation of decisions and the process of choosing one from the mass. The more conspicuous the choice options and their quality, the quicker and better the user decision.
- **Options** are the immediately recognizable choices to resolve a matter.
- **Agendas** are the longer term goals that help influence more immediate decision making.

Act is the translation of the decision into the physical action. This is the slowest component of the loop and is inevitably lagging and more difficult to correct.
- **Results** are (possible) verification to the OODA loop.

2.0 Navigation

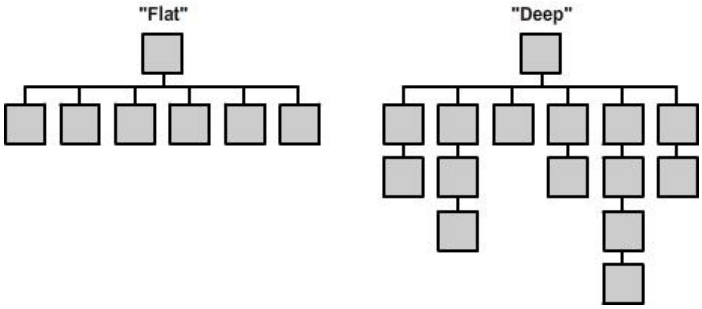
The term navigation in this context refers to the user movement through the interactive product, be it an application or a game. The general concept of navigation is first reviewed and then a break-down of navigation components (screens) is provided for both applications and games.

2.1 Navigation Models

Sequential -- This is a system of navigation, as the name implies to guide the user experience in a linear (or largely) format. It is particularly effective when the user supplies information or makes choices that layer one atop the next towards a desired effect (such as choosing a product or service, editing or specifying it, supplying delivery/mailing information and then purchasing it, in that specific order). Simple applications employ this architecture, but rarely is it otherwise used.

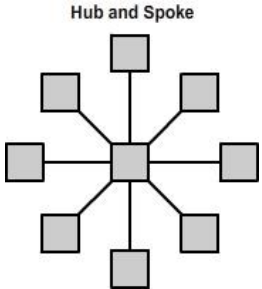
Branching -- This is a system of navigation is a more sophisticated variation of the previous (Sequential) in that the user follows a flow that often has segues ("branches") from a main thread. Wayfinding can be a tricky thing in this architecture so it is of paramount importance to always provide an "escape button" (a means to return to a central or recognizable point). Sophisticated applications frequently employ this architecture.

Flattened Hierarchy (also Flat Hierarchy) -- This is a system of connections arranged on one or more "horizontal layers" to facilitative transparency and ease of navigation. The key characteristic to this format is ubiquitous navigation -- the user is afforded the choice to navigate to anywhere from anywhere (within the "primary tier"). This architecture is common in applications (including most websites) and most games.



> Flat -- This arrangement is the simplest format in that it has only one "depth" of navigation, so it is impossible to get lost (for the user) and very easy to construct (for the developer). The format is also limiting in that it restricts the scope of content options due to the simplistic navigation tools (a smaller number of buttons or tabs are available for practical use). There is no specific number limit of options, but the developer should be mindful to not overwhelm the user.

> Deep -- This arrangement provides the general ease of use provided by a flattened hierarchy but also allows a far greater scope. The format also requires the developer to facilitate the user navigating that larger scope, typically by providing the options to navigate "home" (return to main) and "back" (return to the last visited



location) and possibly more (targeting specific prominent destinations).

Hub and Spoke -- This is a system of connections arranged like a "wire wheel" in which all traffic moves along "spokes" connected to the "hub" at the center. The key characteristic to this format is that the user must return to a central location between navigating to any given "spoke." This architecture is used for applications and games, typically for ease of use (novice users) or for the specific purpose of directing the user to a key location (the "hub") for some specific reason -- commonly for updates, advertisement, updates or the like.

The Hub and Spoke navigational format can also support the option to be "flat" or "deep," drilling down along any "spoke" for given purposes, but the user must still return to a central location before navigating to another "spoke."

2.2 Screens and Windows

Screens -- These, as the name implies, are full-screen displays.

Windows -- A display that is smaller than the screen size and lies atop another display is known as a window. These are frequently employed for temporary use and typically expand on options available from the display below.

2.3 Games General Screens

Splash/Load -- This is the opening "handshake" for the product, traditionally a series of screens introducing the publisher (or class studio) and development team(s) (your team) with names and/or logos. This might time-out and automatically navigate to the Main or might require player input ("press space to continue").

Main -- This often serves as a central hub to navigation and also typically serves to support the "character" of the product.

Options -- This typically provides control over display (resolution options, gamma, etc), audio (sound levels, music/effect disables, etc) and controls (key-mapping, controller-mapping, etc).

Credits -- This provides more detailed information on the development studio and possibly the development team.

Instructions -- This allows new players to familiarize themselves with the controls and possibly play hints. Sometimes this is supplanted or augmented with a tutorial system.

Game -- The game itself…

Results -- Often times the game results are provided with option to replay, or in the case of multiplayer games a chat mechanism so players can talk smack afterwards.

2.4 Applications General Screens

Splash/Load -- This is the opening "handshake" for the product, traditionally a series of screens introducing the publisher (or class studio) and development team(s) (your team) with names and/or logos. This typically times-out and automatically navigates to the Main.

Main -- This often serves as a central hub to navigation and also typically serves to clarify the purpose of the product and promote the developer. Dynamic updates (if any) are often available here.

Help (Instructions) -- This empowers users by providing instructions as needed and often also provides links for technical support. A common practice is to provide a "site map" for sophisticated architectures, and in so doing providing the user with navigational reference.

About (Credits/Contact) -- This provides a summary of the product purpose, legal notice, contact information/links and possibly information on the developer(s) (studio/team/individual).

(Content) -- Depending on the purpose/use/needs of the application there could be one, few or many screens.

3.0 UI prototyping

Just like any other product, success depends largely on thorough planning and testing, adjustment and refinement -- testing and the revisions it fuels are the most important aspect with regards of function and quality. Prototyping UI is accomplished by a number of methods suited to differing results (Fidelity and Prototype Models).

3.1 Fidelity

The word fidelity refers to the adherence to the final quality of a product (such as high fidelity audio referring to the quality of the sound). In the case of prototypes the fidelity roughly expresses how close an approximation of quality to the final product. There are three general tiers of prototype fidelity, each distinguished by the investment required to produce and the quality of the return (meaning the depth of validity gained).

- **Low fidelity prototypes** are characterized by low investment costs (time, effort and capital) that yields a rough value but allows for quite significant iteration. There are a wide range of low fidelity prototypes including sketches, diagrams and demonstrable or even playable paper models. These models are typically (almost always) constructed with practical materials (art or office supplies rather than on computers). See **Paper Models**.
- **Medium fidelity prototypes** are characterized by modest investment that yields a closer approximation to the final product. These are typically computer versions that are operational though lacking the sophistication/polish of the final product. Medium fidelity prototypes often employ middleware to reduce investment and increase yield.
- **High fidelity prototypes** are characterized by great investment for the specific purpose of yielding a very close or the exact quality of the final product, though in greatly reduced scope. These are typically employed as marketing tools since "untrained eye" might mistakenly discount a model that is "rougher". These are commonly called POC (Proof Of Concept) or Vertical Slice (for more technical-oriented models). High fidelity prototypes could employ middleware or existing proprietary game engines, or could be built from the ground up (though engine development is costly).

  High fidelity prototypes are commonly constructed in the intended target environment, called "native" builds.

3.2 Prototype Models

Paper Model -- low fidelity -- the use of "pen and paper" or paper model prototypes ensure reasonable return with a very low investment. The accuracy of the prototype is far from the finished product, but the short development time and accordingly rapid and numerous iterations provide the opportunity for massive exploration and refinement.

Digital Model -- medium to high fidelity -- typically by use of middleware, digital prototypes provide far more accuracy to the final product (that paper models) but at the significant cost of time and materials (including hardware, software and the availability of the skilled individuals using it).

Native Model -- high fidelity -- developing on the native platform again boosts the accuracy to final product but also requires more investment (than medium fidelity middleware prototypes).

These videos from Google review each of these prototype models:

Rapid Prototyping 1 of 3: Sketching & Paper Prototyping
http://www.youtube.com/watch?v=JMjozqJS44M

Rapid Prototyping 2 of 3: Digital Prototyping
http://www.youtube.com/watch?v=KWGBGTGryFk

Rapid Prototyping 3 of 3: Native Prototyping
http://www.youtube.com/watch?v=lusOgox4xMI

3.3 Paper Models

As the name implies, these prototypes are constructed of paper (or posterboard) that typically are cut and drawn to represent aspects of the game (environments and characters, etc). The low technology threshold and ease of construction of paper models serves as the perfect vehicle for rapid prototyping. They are a foundation for early design because of the capacity to iterate and refine.

The paper model is more often a demonstration (like puppet show) rather than a playable construct. It allows the presenter to reveal the concept to others that can then challenge (test) or suggest modifications to the concept. This is typically done amongst peers and frequently various members of the group will present their version on a theme so that the best ideas can be accumulated rather than one idea targeted. Of course with iterations the ideas fuse together.

Benefits:

- **Rapid prototyping** allows for numerous iterations on the design and so more capacity for exploration and refinement.
- **Low cost in time and materials** facilitates the rapid prototyping cycle.
- **No technology requirements** allow participation by members of the design team that do not have programming skills.

Limitations:

- **Inaccurate depiction** of the product requires translation to final product.
- **Interaction is limited** (typically) to talking about the concept (a verbal exchange rather than a tangible/visceral experience).