

Approach to Interactive Application Design

By Matthew Overton

Introduction	3
System Analysis	4
Development Methodologies	4
Waterfall	4
Agile	4
Suitable Technology	5
Desktop Application	5
Web	5
Mobile	5
Augmented Reality, Virtual Reality and other future technology	6
Interaction Models	6
Cognitive Walkthrough	6
Psychology of Interactions	7
Recommendations	8
Bibliography	Error! Bookmark not defined.
Appendix	11
A: Chess flow diagram	11

Introduction

As the number of people who have access to a computer increased, the opportunity for online social interaction did also. It's human nature to crave social interaction from others and with social networking giants like Facebook, this innate urge became a reality very quickly. While Facebook's primary function was to bring worldwide communication to your fingertips in the shadows a large scale gaming infrastructure was forming. Because this formation was a byproduct I feel like the interaction within facebook games is extremely lacking and could be improved upon immensely when creating the application for D20 Games.

The most efficient way to develop an interactive application is to look at the most important features and flesh out those first. You can then build upon that foundation with an infinite amount of content[1]. One of the great benefits of playing board and card games on an application is the level of complexity can be increased massively. There is no chance to lose a rulebook or make illegal moves and mistakes because the computer handles the rules for you, you just play. As someone who frequently becomes the person to read and establish the ruleset to others it's refreshing to just play a game out of the box and learn as you go[2].

A classic and well known board game to adapt would be chess. Most people can recognise chess upon first glance due to the iconic checkerboard playing field and black and white pieces. It requires two players, human or AI, and typical games last around 40 moves; the flow of the game can be easily identified by three phases: Opening, Middlegame and Endgame. Chess can appeal to a wide audience signified by the large age gaps between grandmasters ranging from 12 to 97 years old. It is even recommended to teach a child chess from as early as five years old[3]. Chess is beautiful in it's complex simplicity but adapting chess to an application opens up a range of tools to increase its accessibility and popularity within children especially. The rules of chess are very well documented which makes an adaptation much simpler, an easy approach would be to develop the application for the web because it's the most widely available and has in built networking capabilities necessary for a multiplayer game. An important aspect of chess is the feel of taking the other players pieces physically which can be lost when developed in a virtual environment, in order to preserve this feeling you could create unique animations for taking a piece that feels satisfying.

Bridge is card game which heavily favours the interaction between a team of two during a bidding phase within the game[4], it's very difficult to replicate over an application due to the computers limited ability to show emotions. Using newer technology such as virtual reality (VR) this becomes more plausible, it makes it easier to replicate the subtleties of human interaction and helps imitate how the players would feel in the real world.

System Analysis

Development Methodologies

Waterfall

Andrew Powell-Morse - *"The waterfall model emphasizes that a logical progression of steps be taken throughout the software development life cycle (SDLC), much like the cascading steps down an incremental waterfall"*[5].

The most logical development method comes from the waterfall method. Working in clear steps, it's easier for a development team to stay focused and complete everything that is necessary before moving on to the next step. The waterfall method usually doesn't allow returning to a previous step so it values spending more time fully investigating a step before moving on.

Agile

Synopsys Editorial Team - *"Agile methods attempt to minimize risk (such as bugs, cost overruns, and changing requirements) when adding new functionality by developing the software in iterations that are mini-increments of the new functionality."*[6].

There are many variations of the agile development methodology but they all comply to the same basic principles. At the beginning of each development cycle analyze the application to structure the features in order of most importance, then complete as many of those features as possible in the current iteration. The benefit of the agile methodology then comes into fruition because the updates are deployed and the next cycles order of importance is dictated by the user feedback; It provides essential developer-user communication and places more importance on the user experience.

For D20 games I believe a combination of both waterfall and agile methods should be used to maximise the potential of a physical to online board or card game. The initial conversion should use the waterfall method to clearly identify what makes the original game popular and beloved using the more detailed system analysis. This will maintain the integrity of the original game while opening up the possibility of improvements made by the community.

Suitable Technology

There are so many options when it comes to developing new applications and the technology and language you choose is specifically tailored to the requirements of the program you are developing.

Desktop Application

Desktop applications are the traditional approach to creating programs. They utilize the hardware you have installed on your machine for performance so users without the minimum specs may find it difficult to run the application[7]. Once installed however desktop apps can be run at all times even when no internet connection is present, unless some form of internet verification is required. Some common languages for desktop applications include Java, C++, C# and python. While desktop apps have been slowly declining there is still demand for them due to the reduced hosting cost and robust object-oriented languages.

When it comes to user interaction, desktop applications are encouraged to stick to the user interface guidelines enforced by the operating system the app is being run on. An example of this is how all apps on MacOS follow the same clean, minimal look that Apple presents in all of their products, Microsoft offers more flexibility with their customization on Windows devices[8] and then Linux is totally open-sourced and all interaction is dictated by the developer. It is important to note that when designing a UI for a desktop application there should be as little steps as possible to reach the specified command, three being the absolute maximum[9].

Web

Web applications have been on the rise over the last decade because they are accessible on any device within the browser using a common scripting language such as JavaScript. This makes it far easier to reach a wide audience due to the lack of hardware and operating system limitations. Web applications offload their computing power to the server they're being hosted on[10] and so as the popularity of a web application escalates it becomes increasingly more important to boost the power of the web hosting leading to increased costs.

For user interaction, web applications only need to follow general internet guidelines. There is a dedicated styling language, CSS, which developers can use to make their applications stand out and with the help of some JavaScript add some hover effects, transitions and more. Using modals and opening new tabs within the browser users can still multitask with minimal effort within a web app.

Mobile

Mobile applications were an important step to increase the popularity and useability for applications. When the first iPhone was released alongside the App Store it became apparent that carrying around the functionality of a computer was possible by using mobile applications. Like desktop applications the only limitations are the devices hardware capabilities.

Mobile devices offer a unique form of interaction in that instead of an extra peripheral such as a mouse, controls are accessed via touch. This allows the user to get direct haptic feedback to their actions and offers more versatile ways to improve user experience.

Augmented Reality, Virtual Reality and other future technology

Augmented Reality (AR) and Virtual Reality (VR) are up and coming technologies that primarily focus on bridging the gap between the real world and virtually generated ones.

Augmented reality is the less demanding of the two, requiring only a camera connected to either a computer or more commonly a smartphone. An excellent example of how to use AR technology is in Niantic's Pokemon Go, this game places pokemon in the real world using the smartphone's camera and allows you to catch pokemon with a throwing gesture using the smartphone's accelerometer[11]. It makes a compelling attempt to allow virtual creations to interact with the real world and in my opinion succeeds in a new form of user experience encouraging exercise.

Virtual reality is one of the more promising technical feats of the last few years, showing that it is truly possible to immerse yourself in a totally virtual world and interact like it's real. Companies such as Oculus and HTC have prioritized user experience by making the specialized headsets as light as possible and their controllers ergonomic. Virtual reality offers a unique experience that no other technology can at the moment, it truly replicates the physical interaction of the real world inside of an application. VR works due to a combination of simulating how the real world works along with your brain filling in the gaps from past experiences and instinct[12], this allows a user to be transported into a new and exciting world without feeling motion sick or confused.

Interaction Models

An example of how a chess match may play out from start to finish can be found in Appendix A, it shows how the user would select a side, choose the difficulty of the AI and then play until a winner was found. Alternatively the user could search for another user online with the same ELO rating.

Cognitive Walkthrough

D. Norman, et al. in 2013 in "The Design of Everyday Things"[13] talks about how users interact with an interface. He proposed that there were seven stages to a user interacting with an interface.

- Establishing the goal
- Formulating intention
- Specifies action in the interface
- Executes the action
- Perceives the state of the system
- Interprets the state of the system
- Evaluates if the system state matches the goal

Using this model you can design user interfaces intuitively by matching what the user would like to do with how your interface is laid out in front of them.

A cognitive walkthrough follows the same principles of Donald Norman's seven stage model, it is a usability inspection method used to identify problems with usability within an interactive application, specifically how new users will interact with the system that has been designed. Cognitive walkthroughs are beneficial because they are less costly than a full usability lab study but have the capability of identifying potential problems with a user interface before coming into contact with a user.

Blackmon, Polson, et al. in 2002 in their paper "Cognitive walkthrough for the web" indicate four questions to be used during a cognitive walkthrough[14]:

1. Will the correct action be made sufficiently evident to the user?
2. Will the user connect the correct action's description with what he or she is trying to do?
3. Will the user interpret the system's response to the chosen action correctly?

As an example I will fill out a cognitive walkthrough for a chess application on the web. Assuming the game functions like the classic physical chess game and the human player is the white pieces.

Goal: Make your chess move

1. Yes, clicking a chess piece is a natural parallel to picking one up.
2. Yes, clicking a chess piece will raise it from the board and show available spaces.
3. Yes, after clicking the final location the piece moves and moves onto the other players turn completing the goal.

Goal: Resolve check

1. Yes, yellow indicator will show that the king is in check.
2. Yes, only moves that resolve the check can be performed.
3. Yes, after resolving the check the yellow indicator goes away.

Psychology of Interactions

The purpose of the user interface is to provide your brain enough stimuli to make meaningful actions toward a goal, using psychology we have been able to determine how to keep the attention of the brain in order to improve user experience[15]. It is important to try and incorporate the '*cocktail party*' effect[16], the idea of singling out a voice while simultaneously dulling all of the others at a crowded party, inside an application in order to direct the user by using colours, fonts, white space and audio cues.

Another psychological concept to keep in mind when designing a user interface is Hick's Law, which states the time it takes for a user to make a decision logarithmically increases based on the number of choices presented[17]. Actively limiting the options the user has speeds up the decisions dramatically and improves user experience.

Designers are encouraged to support a state of 'flow' when designing their user interface, coined by psychologist Mihaly Csikszentmihalyi in 1975 'flow' is a state of fully focused immersion induced by participating in an activity that is both high skill and challenge. Chess is a game which induces a state of 'flow' and the UI should be direct enough to not break that focus.

Recommendations

D20 Games have many routes they could take when it comes to adapting a game like chess into an application. One option would be an AR approach where a user could use a smartphone camera on any chessboard they own to display any number of ongoing chess matches. The user can simply challenge a friend to a game and the piece positions will be stored in the cloud. The application would need to be developed using an engine that supports AR such as unity or the creation of its own engine. This option is very time consuming and the engine licensing is expensive.

Due to the target audience being so large I think making a web application would be the perfect route. Web applications are available on all platforms and chess can be programmed within JavaScript without too much hassle. Emphasis should be placed on creating meaningful feedback when moving pieces such as hovering and piece capturing animations. A direct minimal UI should be present to comply with both Norman's model and maintain Csikszentmihalyi's idea of 'flow'.

Bibliography

1. Silverman D. How to Learn Board Game Design and Development [Internet]. Game Development Envato Tuts+. 2013 [cited 25 February 2019]. Available from: <https://gamedev.tutsplus.com/articles/how-to-learn-board-game-design-and-development--gamedev-11607>
2. What are the advantages of playing online games? [Internet]. Quora. 2017 [cited 25 February 2019]. Available from: <https://www.quora.com/What-are-the-advantages-of-playing-online-games>
3. The Chess Files: At what age can you begin to teach chess to a child? – The U.S. Chess Trust [Internet]. Uschesstrust.org. 2012 [cited 25 February 2019]. Available from: <http://www.uschesstrust.org/the-chess-files-at-what-age-can-you-begin-to-teach-chess-to-a-child/>
4. Understanding the Importance of Bidding in Bridge - dummies [Internet]. dummies. [cited 25 February 2019]. Available from: <https://www.dummies.com/games/card-games/bridge/understanding-importance-bidding-bridge/>
5. Powell-Morse A. Waterfall Model: What Is It and When Should You Use It? [Internet]. Airbrake Blog. 2016 [cited 25 February 2019]. Available from: <https://airbrake.io/blog/sdlc/waterfall-model>
6. Top 4 Software Development Methodologies | Synopsys [Internet]. Software Integrity Blog. 2017 [cited 25 February 2019]. Available from: <https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/>
7. Bychkov D. Desktop vs. Web Applications: A Deeper Look and Comparison [Internet]. Segue Technologies. 2013 [cited 25 February 2019]. Available from: <https://www.seguetech.com/desktop-vs-web-applications/>
8. Shen S. Differences Between Designing a Web App and a Desktop App [Internet]. Muzli - Design Inspiration. 2018 [cited 25 February 2019]. Available from: <https://medium.muz.li/differences-between-designing-a-web-app-and-a-desktop-app-e7d65a7545eb>
9. Lee A. Designing for PC Apps – UX Planet [Internet]. UX Planet. 2018 [cited 25 February 2019]. Available from: <https://uxplanet.org/designing-for-pc-apps-4554d8a0aa85>
10. Rocheleau J. Why Web Apps Are Becoming the New Native Apps - Envato [Internet]. Envato. 2017 [cited 25 February 2019]. Available from: <https://envato.com/blog/web-apps-becoming-new-native-apps/>
11. Warner C. This Is How "Pokemon Go" Actually Works [Internet]. Bustle. 2016 [cited 25 February 2019]. Available from: <https://www.bustle.com/articles/172317-how-does-pokemon-go-work-heres-everything-we-know-about-the-tech-behind-the-augmented-reality>
12. Don't be "tech basic:" How virtual reality tricks your brain into thinking it's real (Paid Content by Lenovo) [Internet]. Mashable. 2018 [cited 25 February 2019]. Available from: <https://mashable.com/article/science-behind-virtual-reality/?europe=true#F69aSAKrWsq7>
13. Norman D. The Design of Everyday Things [Internet]. 2nd ed. New York: Basic Books; 2013 [cited 25 February 2019]. Available from: <http://www.nixdell.com/classes/HCI-and-Design-Spring-2017/The-Design-of-Everyday-Things-Revised-and-Expanded-Edition.pdf>
14. Blackmon M, Polson P. Cognitive Walkthrough for the Web [Internet]. 1st ed. Minneapolis: University of Colorado; 2002 [cited 25 February 2019]. Available from: <http://autocww2.colorado.edu/~blackmon/Papers/BlackmonEtAlCHI2002.pdf>
15. Gaal N. UX & Psychology go hand in hand — Introduction to human attention [Internet]. UX Collective. 2018 [cited 25 February 2019]. Available from: <https://uxdesign.cc/ux-psychology-go-hand-in-hand-introduction-to-human-attention-a70ffd2c4289>

16. How selective hearing works in the brain: 'Cocktail party effect' explained [Internet]. ScienceDaily. 2012 [cited 25 February 2019]. Available from: <https://www.sciencedaily.com/releases/2012/04/120418135045.htm>
17. Soegaard M. Hick's Law: Making the choice easier for users [Internet]. The Interaction Design Foundation. 2019 [cited 25 February 2019]. Available from: <https://www.interaction-design.org/literature/article/hick-s-law-making-the-choice-easier-for-users>

Appendix

A: Chess flow diagram

