Assignment P3

Michael Tong
mtong2@gatech.edu

Question 1

Discoverability

Discoverability, which is often described with visibility, is one of the key principles for successful interface development. This facet governs a user's ability to interact with the computational interface by providing them information on what can actually be done and often acts as the first source of operational information. As such, this principle directly influences the creation of an invisible interface by controlling the gulf of execution. With the primary characteristic of an invisible interface being that a user doesn't have to think about it in order to achieve their goal, discoverability directly influences this characteristic through dictating the actions that are available to the user, and in doing so alters the gulf of execution by altering the user's perception of what is actually possible.

Simplicity

Similar to the principle of discoverability, simplicity acts to influence the gulf of execution through manipulation of available actions. Often, simplicity works in tandem with discoverability by limiting the actions available to the user but maintaining the ability to accomplish the user's goals and expectations of the interface. Simplicity thus supports the development of an invisible interface by shrinking the gulf of execution in focusing the user's intentions into more concise actions. In essence, the principle of simplicity is the removal of noise, or irrelevant information from the interface but retains the ability to achieve the user's goals.

Affordances

The principle of affordances is related to both discovery and simplicity and supports the development of an invisible interface by hinting to the user what actions are possible. Affordances is a form of discoverability that exploits heuristics in their inherent design and thus supports an invisible interface

similarly. Affordances affect the gulf of execution through using a user's heuristic knowledge of a particular design into limiting the perception of possible actions. The interface becomes more invisible as a result of heuristics because the translation of goals into actions utilize the user's prior knowledge.

Ease

Separate from the user's specific interaction with an interface but similar to the benefits of an invisible interface, the principal of ease should be considered in the participant view of the user. Especially for interfaces that require multitasking, ease reduces the cognitive load and fatigue caused by the interfaces, which more efficiently allows the user to manage competing tasks. Ease emphasizes the participant view of the user through reducing the time needed to think about the interface.

Constraints

In addition to the principal of ease, constraints may also improve the efficacy of the user from the participant view. Constraints can aid by reducing error through preventing the user from performing incorrect actions. In the context of the participant view, other interfaces and tasks may be drawing attention, and the implementation of constraints could alleviate slips and mistakes when transitioning from interface to interface.

Question 2

Throughout my daily life the use of a computer terminal (Ubuntu for this discussion) is intolerant to error. This characteristic arises as a result of user proficiency where experienced user's often find the interface to be the epitome of simplicity, while newer users often consider it the bane of computational interfaces. The interface is simply a black box (literally) that accepts special commands and can be used to control just about every aspect of a computer. With this powerful, general purpose interface, catastrophic errors can be made with

very simple keystrokes which often have no remorse. For instance, in common computer graphical user interfaces (GUIs), if you wish to delete a file, it will ask for confirmation in case you did it by accident, and often temporarily keep the file stored somewhere else just to be sure. In a terminal, it assumes that you know what you're doing and doesn't ask twice (outside of interactive mode). To even more extremes, an entire computer can be deleted in under 15 keystrokes on request, no questions asked. Often, when drastic mistakes are made in a terminal, even the almighty Google will tell you "too bad".

While the purpose of the terminal is to provide users unlimited control of their computer, there is an extremely steep learning curve. To alleviate this trend, an improvement to the terminal interface could be the implementation of different versions that impose specific constraints to account for user expertise. For instance, if a user asserts that they are a beginner, the system could restrict access to system files, where tampering often causes significant issues and are difficult to repair. Other constraints for newer users could be the removal of the ability to recursively force sudo remove or sudo remove in general, which are some of the most unforgiving commands, where folders and files are removed without question. These constraints would aid in avoiding errors for new users, who often spend a lot of time learning through online resources and may blindly follow commands they see.

In addition to constraints, the mapping between action and results can be improved in the terminal interface. One modification that can be implemented is displaying the result of the user's input prior to execution. For instance, if a user inputs a command to move all files that begin with "HCI", the terminal will first display all the files being moved and ask if they are sure they want to move them. This mapping provides the user a better visual of what their command will actually perform, and hopefully catch errors before the user fully executes the command.

Implementation of affordances is also another approach to avoid errors. One such implementation can be the addition of greyed out or transparent/outlined file and directory names when a remove command is made. In the command prompt, the next use of the list directory command could display to the user all the files and directories as well as the recently removed ones as shown in Figure 1 below (I apologize for my computer graphical design ability).

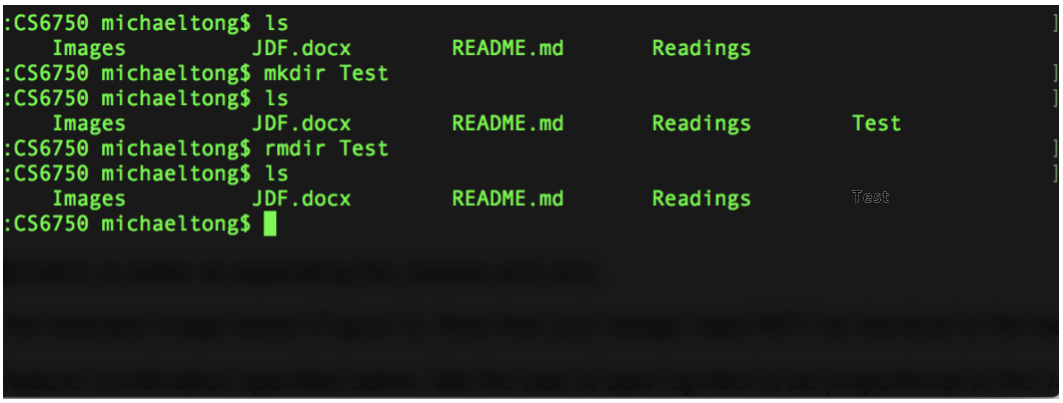

Figure 1: Example affordance redesign to the terminal for removed objects

All of the redesign examples discussed would mostly be tailored towards entry level users who may not understand the true expanse of their commands. The discussed constraints may prevent the execution of simple, yet devastating commands, and mappings and affordances can better visually display to the user their actions, and hopefully catch errors before execution.

Question 3

In the game of chess, slips and mistakes are made often regardless of the proficiency of the individual. Slips are common when individuals are fixated on a specific plan on action and may leave themselves vulnerable to an unexpected counter attack. In contrast, mistakes can occur when players aren't fully aware of rarely experienced situations in the game, such as a when a pawn moves across the board and is "queened".

Slip

The game of chess is unique in that it requires constant reevaluation of the user's plan of action, which requires a significant amount of knowledge of the game. A common memory lapse slip that amateurs make is they tunnel vision on

a particular plan and forget about the movement of their opponent's pieces. For instance, player 1 may be planning to trap their opponent's queen with a few moves, but after doing so, leaves their king open for a checkmate because they didn't factor in the moves that the opponent made during their execution. The player knows not to place themselves in a position where a checkmate is inevitable but is often too focused on their offensive to realize their mistake until it's too late.

In the modern era, a training interface can be created to notify players of potential mistakes, such as a move that would leave them exposed to a checkmate. While using this interface, when a move is considered, but the system determines that the other player will be able to checkmate them in the next move as a result, the interface could ask the user if they are sure they want to make the move and display to the user the move that the opponent can make to arrive at a checkmate.

Mistake

Unlike the memory lapse scenario discussed in the event of a slip, a mistake that may occur with newer players are the less common scenarios that may arise in some games but not all. Most players are aware of the general moves available to each piece at a given time, but many new players probably don't experience a situation where a pawn is queened, which occurs when a pawn crosses to the opposite end of the board and is allowed to become promoted (formal term) into any other piece except a king (often a queen, hence the term "queening"). The mistake in this situation is when a player is unaware as to why their opponent is moving their pawn forward and doesn't stop the charge until it reaches the other side and then learns about the rule.

A potential interface improvement could be the creation of a chess board that displays some tips and tricks to the game etched or placed on both sides of the board. The panel would then display the rule of queening to the users to make them better aware of the niche rules to the game.

Conclusion

The game of chess is challenging due to the number of variables and degrees of freedom at each state of the game. Masters of the game often have a considerable amount of experience which mostly comes in the form of memorizing the position of pieces at a particular state as well as the optimal moves moving forward. Throughout the game, the optimal move, if there exists one, is almost never obvious since it relies on the movement of your opponent. Due to the virtually infinite number of possible game states, the ability to plan forward effectively is often limited to a few turns.

Question 4

Good Interface

The Canvas interface provides a good representation for displaying relevant class content to the user. The interface's goal is to provide a gateway for users to access classes and the home dashboard is both clear with visual representations, as well as limiting extraneous information. To emphasize the characteristics for why Canvas is a good representation for its content, the interface contains multiple forms of visual information to aid the user in determining which tile belongs to which class. The first form is the image, which is unique to the class and often resembles the purpose of the class in some manner. The next visual aid is text, where both the class ID and class title are displayed. Conjunction of these two visuals makes the representation of the class more salient to the user.

In addition to having multiple visual cues to better represent the relationship of tiles to a specific class, another reason for Canvas being a good representation is that it excludes extraneous details. Relevant information to the user is displayed on the front page. Most students only care about their upcoming assignments and classes, and Canvas does a beautiful job in displaying this information with the classes as tiles on the left, and the user's assignments on the

right, while not removing additional features, which are placed on a smaller toolbar on the left.

Poor Interface

In contrast, it can be argued that T-Squared, the predecessor to Canvas, provides a poor representation of its underlying content. Both interfaces shared the same goal of being a hub for students and their classes, but T-Squared provided visual ques through text and the class titles were not used, only the class ID is shown. Additionally, the front page contains a lot of information for announcements and messages which makes the dashboard feel more like a social media home page than a location for class information.

The first violation to the T-Squared interface being a good representation of its purpose is that relationships are not explicit. The tabs at the top bar representing classes does not clearly define which class is being represented and often leads to memory lapse slips. Since only the class ID is shown, and these consist of a string of slightly random characters, users may mix up one class ID with another and accidentally select the wrong tab.

To further describe why the T-Squared interface is a poor representation of its content, the homepage fails to exclude extraneous details. As discussed with Canvas, most users are interested in their class homepage and their upcoming assignments. The T-Squared interface instead displays an elaborate arrangement of announcements, messages, and a calendar. This information provided by these sections are often irrelevant, messages never appear, the calendar does not auto-populate like Canvas', and the announcements are compiled across all classes, which often clutters the information.