CS6750 - P3

Lily Pytel
lpytel3@gatech.edu

1 Question 1

1.1 Invisible Interfaces

1.1.1 Mapping

Mapping is a key component to creating visual interfaces because it is centered around making interfaces that are similar - at least map seamlessly - to the content they represent and make use of well-known concepts, such as buttons and sliders. For example, mapping a tap to a button click simulates the physical activity of pressing a button on a device such as a calculator. This principle bridges the gulf of execution by making the *specify* stage easier to accomplish.

1.1.2 Tolerance

The principle of tolerance allows a user to easily exit an unwanted state. To make an interface invisible, the user must be able to exit in and out of states without thinking about it. If they have to get themselves out of a bind using a series of complicated actions, they will have to focus on the rules of the interface instead of having the interface operate according to their rules. Thinking about the interface instead of the task harms its invisibility. On the other hand, if a user can go back and forth between actions seamlessly, they will be feel more comfortable using the interface naturally. This bridges the gulf of evaluation by helping the user *perceive* the state of the system. One could also argue that it bridges the gulf of execution by making the *perform* stage more seamless.

1.1.3 Consistency

Once an interface is learned, the user can stop perceiving it and focus only on the task itself. However, if parts of the interface were inconsistent and required additional learning or remembering edge-cases, it increases the cognitive load on the user and forces their attention away from the task. Making interfaces consistent bridges the gulf of execution by simplifying the *plan* and *specify* phases. If the user does not have to think about how to do things, they can focus

on planning and specifying for the actual task, instead of planning which actions they need to make on the interface.

1.2 Participant View

1.2.1 Equity

The principle of equity states that interfaces should accommodate a wide range of individual preferences and abilities. By definition, that means taking into account the context of the user - specifically, their knowledge, preferences and abilities, which all exist outside the interface itself. To design an interface following the principle of equity, we have to focus on the types of people (in this case, their contexts) that would use it. For example, a user with the context of having born with smartphones would have different abilities than a person who was born before smartphones.

1.2.2 Ease and Comfort

The principle of ease and comfort states that an interface should be used comfortably and cause the user the minimum amount of fatigue. Different contexts might have different definitions of comfort and different sources of fatigue. For example, constantly swiping on a touchscreen eventually causes chafing, whereas constant mouse movements eventually cause wrist strain. To design interfaces with this principle in mind, one must take into account all contexts in which the interface will be used.

2 Question 2

2.1 Interface Intolerant to User Errors

Mint[1] is a website that allows users to track their spending, create budgets and set financial goals. Specifically, there is a budget page where a user can set a budget for different categories for the current month. Only the current month can be set at a time. I will focus on an ability that is not often advertised, which is the ability to rollover debt or a surplus in a budget across months. For example, if I budgeted $100 for shopping and only used $50, I can carry over $50 to next month. The ability gets enabled via a checkbox when setting or updating a

[1] https://mint.intuit.com

budget. Figure 1 shows the window for creating a new budget, with the checkbox already checked.

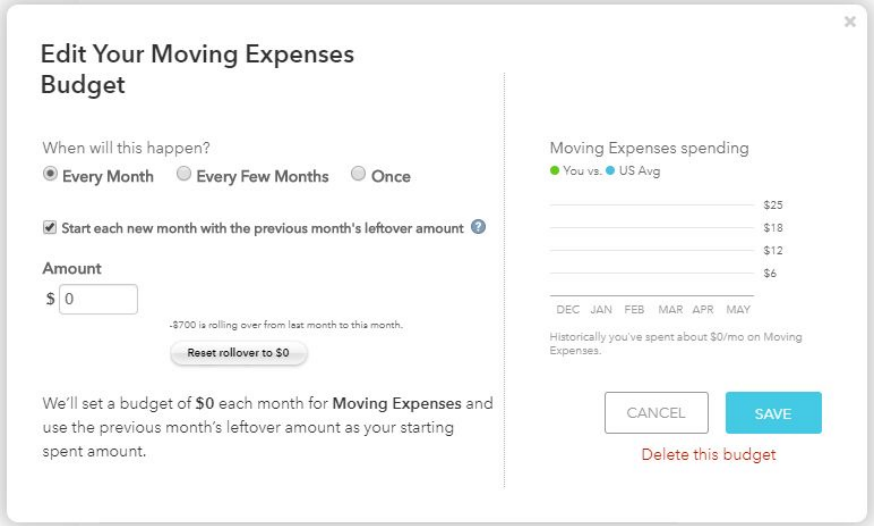

Fig 1. The interface for setting a budget in Mint.

In general, Mint does not show any warnings before performing actions. When it comes to creating new budgets, that fact is not important. However, when it comes to deleting budgets or making changes, there is no way to revert the change and no warning about the repercussions that are about to happen. For example, figure 1 shows a budget with a surplus of $700. That means that $700 have been saved in previous months for moving expenses. If the user clicks on "Delete this budget" or "Reset rollover to $0", the surplus will be lost with no possibility of recovery.

This error can be done as a slip or a mistake - the link "Delete this budget" is not far from the Save button, so it would be easy to click it by accident - this would be an *action-based* slip. In addition, if a user does not know that the surplus is not recoverable, they might delete the budget, thinking they can set a manual surplus, thus making a *Knowledge-Based* mistake.

2.2 Constraints

Without attempting to redesign the entire interface of Mint, constraints could be used to at least warn the user when they are about to perform an action from which there is no return in the form of a confirmation dialog. Another option

would be the ability to set the rollover to anything as opposed to just "Reset rollover to $0".

2.3 Improved Affordances

Improved affordances could be used to avoid errors by introducing a feature that allows users to create multi-month budgets that accounts for funds saved during previous months. To add an affordance signifier, we could make a multi-month budget line longer than others or add a familiar icon that denotes that something has more content (such as three dots or an arrow indicating something is collapsed and could be expanded).

2.4 Improved Mapping

Improved mapping could be used to avoid errors by showing the user what would happen if they reset the rollover to 0, thus making the consequences of their actions more visually obvious. For example, have a progress bar (an element that is commonly used in software) to denote the rollover and allow the user to move it in order to set it manually to the number they desire.

3 Question 3

The game I have selected is Sid Meier's Civilization VI, a computer game in which the user builds an empire across multiple milenia. Players can build cities, districts within cities, mines and world wonders such as the Alhambra. They can also war with other players.

3.1 Slip

One slip that I commit fairly often is to build the wrong district in a city due to a wrong or clumsy click. I might have meant to make a financial district and accidentally clicked on the district above it. This is an *action-based* slip and does not involve any lapses in memory - I know what I wanted to do, but my finger slipped on the mouse. Sadly, the game has no undo mechanism.

One seemingly simple way to remedy the situation is to provide an undo capability. Note that this is a popular request that has never been fulfilled, to the point that it seems to be by design. Undoing a move after seeing its consequences could be unfair and defeat the purpose of making decisions with grave consequences. Regrets are a healthy part of any empire.

3.2 Mistake

Declaring war in Civilization VI is a gamble. As mentioned above, there is no way to undo the action, nor is there a way to achieve peace right away - the war must continue for some minimum number of moves. A player may want to accomplish world domination, but make the mistake of declaring war on a civilization they are not equipped to beat. They will usually make this mistake because they underestimate their opponent's army or overestimate their own. This would be a *knowledge-base* mistake, because they would be lacking the knowledge to make a good decision. In the case of this game and many others, the interface hides information on purpose so as to encourage mistakes, since they are part of the fun and challenge of playing.

In this case, this mistake is an important challenge of the game, so the makers of the interface would most definitely not want to warn against it or make it impossible to make - quite the opposite. However, for educational purposes (such as a tutorial), the game might respond to a war declaration by displaying a message about the dangers involved - for example, "England has twice as many units as you. Are you sure you want to declare war?"

3.3 Challenge

The game has several win conditions, one of them being having the most advanced science than any other civilization. The challenging aspect is to figure out how to make your civilization more scientifically advanced than any other while still keeping track of all other aspects of the game. If a player were to only build universities, they would leave themselves without an army, open to attack and would soon lose their edge as other civilizations destroyed their cities. The challenging aspect of the game is balancing the need of the empire while striving to get ahead of all other opponents.

4 Question 4

4.1 Good Interface

I chose WhatsApp as an interface that has a good representation of its underlying content. WhatsApp is a chat application that exists both on smartphones and in the browser, but I will focus on the smartphone application, specifically the main page, as shown in figure 2.

Fig 2. The homepage of the WhatsApp application.

4.2 How Does the Interface Exemplify a Good Representation?

This interface exemplifies two of the characteristics of a good interface: it brings objects and relationships together and it excludes extraneous details. The objects on the homepage screens are images of contacts, contact names, last message snippet, last message date and whether there are any unread messages. The interface groups the objects together in a way that makes the relationship between them clear - for example, each contact image comes to the left of a contact name, signaling that the image corresponds to the name. The last message snippet is placed in between a contact name and a border, indicating that is it is the last message from that contact. Lastly, the date of the last message or number of messages unread is placed at the right of each contact, signaling that they belong to the contact next to which they are placed.

4.3 Bad Interface

I chose the cruise control menu on my Honda Fit 2013 as an example of an interface that has a bad representation of its content and capabilities. Cruise control is a feature that allows drivers to set their car speed to be consistent without having to press on the gas pedal. When going on a highway, you would turn on cruise control to give your foot a break from constantly pressing on the gas and maintain a consistent speed. Figure 3 shows the interface of cruise control as it appears on the Honda Fit 2013.

Fig 3 . The cruise control panel on the Honda Fit 2013.

4.4 Mismatch Between Representation and Content

In essence, cruise control can be started and stopped. While it is started, the driver can accelerate and decelerate their speed without touching the gas pedal, and the speed will remain as they selected. However, the display in figure X does not represent those abilities.

Firstly, the interface does not make the relationship between objects explicit. Laying out the CRUISE and CANCEL buttons at opposite ends makes it seem like CRUISE makes cruise control start and CANCEL makes cruise control stop. The names of the buttons also suggest that. However, in reality, the driver has to press CRUISE and then DECEL\SET in order to start cruising. The CRUISE button simply makes the car "ready" for cruise control. I suspect it is in case the driver presses it by accident, but I am not sure. The CANCEL button does indeed stop cruise mode, though it does not take the car out of "ready" mode - for that, the user has to press the CRUISE button again. To make the relationship explicit, it would be better to have a start and stop button, and a deceleration and acceleration flip.

Secondly, it does not bring objects and relationships together. We can infer that SET and CANCEL have a relationship to each other based on their name, but not their location. Conversely, the SET and RES buttons are close together in a way that suggests they have a relationship. When cruise control mode is on, they do have a relationship because they become the ACCEL and DECEL buttons. However, when cruise control mode is off, there is seemingly no relationship between RES and SET, despite their location. In fact, I cannot find documentation explaining what the RES button does on my particular make and model. All I know is that it is apparently RES for "resume" and not "reset", as I guessed.

Unless I am mistaken and the button serves an important purpose, one could claim the interface also violates the characteristic of removing extraneous details by including a label for a button that is not intuitive and clearly unnecessary for using the feature.