Tyler Small

Dr. Sabarish V. Babu

HCI – CPSC 4140/6140

February 22, 2019

**The Psychology of Everyday Actions**

**Report Introduction**

**Chapter Introduction**

**The Gulf of Execution**

When people use an object or device, they first face what is called – the Gulf of Execution. Norton defines this as the period in which a user is attempting to figure out how the device or object operates. The role of the designer is to design devices that afford a user to execute the device in the correct way and have interfaces that respond in the appropriate manner for the action taken. In other words, the gulf of execution should not be unpleasant when reflected on in the Gulf of Evaluation.

The Gulf of Execution is significant in design because it is the first impression a user will have on a product. Getting to the evaluation period as quickly and painlessly as possible is what most people will perceive as good design. Spending too much time in the Gulf of Execution will likely lead to frustrated, unhappy users.

**Example**

**Positive:**

Visual Studio Code. It has an interface similar to Visual Studio, so it is easy to learn for users of IDEs, and it uses tabs to hide a lot of the controls – making it accessible for new users. The tool is easy to learn for all users and minimizes time in the Gulf of Execution.

**Negative:**

Linux Terminal. It gives no information on how to use it. All users are forever in the Gulf of Execution – i.e. having to look up commands on how to use it.

**The Gulf of Evaluation**

“The Gulf of Evaluation reflects the amount of effort that the person must make to interpret the physical state of the device and to determine how well the expectations and intentions have been met.” (Norton 39). The gulf is minimized with good feedback and a well thought out conceptual model. The idea of designing in accordance to the Gulf of Evaluation is designing in a way that matches the user’s expectations of the app and its controls, with the actual implementation – at any point in time.

The Gulf of Evaluation is an important tool to design around because it is the connection between the user and the device. Spending too long in the gulf is most likely because the user has to think too hard about using the device – and hence creating a bad relationship.

**Example**

**Positive:**

GCC compiler with the Clang Extension. You always know the state of the code you are compiling via detailed error messages that include line numbers. It is not all that often that the compiler is at fault for your confusion or being lost as to what to do next – rather it is the complexity of the language.

**Negative:**

Trying to implement anything into Unreal Engine. The code uses macros in every direction and there is no affordance as to where to find anything in the code. You are forever lost and confused trying to navigate the state of the engine code as a user who did not write the code. And changing anything almost always does something you did not expect.

**Figure 1.** An example of a spellcheck error in a word processor. Affording the user to take an action.

**Human Cognition and Emotion with Implications to Computing Systems**

**Visceral**

The visceral response in concern with interaction design is the immediate emotion the user feels when using a product. The visceral response is not dictated by reason or logic, but purely off of biological response. The biggest take away from understanding users, and their visceral response in ID is in designing user interfaces. The user interface of a product should invoke a positive visceral response … whether that be by use of colors or layout of buttons, the user should – without thinking, be able to decide that your product is inviting and well made.

**Behavioral**

Behavioral response is the reaction to an event, that not necessarily need be thought out. This is the stage where a user moves towards completing their goal, without necessarily planning the steps. This is where good Interaction Design comes to play, a good design should aid the user in behaviorally reaching their goal. Help them not think – but just do. This can be achieved with proper use of feedback, proper feedback helps the user feel in control. And understand where they are in the device, without having to trace back steps. A good design incorporates feedback that caters to a user’s learned behavioral states.

**Reflective**

A reflective response is a state that happens with the most thought and cognition. It is slow, and often occurs after the events have happened. With the thought brings emotion. This is where your experience as designed from an interaction, will be reflected on. The idea around designing towards reflective thought – is to leave the user with a train of past events that invoke positive emotion. This can be done by avoiding things that may lead to guilt or frustration, especially at the ending stages of use.

All three responses are tied together, and important to good interaction design. Immediate actions should lead to proper responses, actions which require work should reward with gratification. Inputs should get responses, so the user can leave the experience feeling in control, and in good spirit when reflecting on past use of the design.

**People as Storytellers**

“People are innately disposed to look for causes of events, to form explanations and stories” (Norton 57). Norton uses the idea of a story as a medium for users to chain series of events into results. This is where he ties the idea of conceptual models as a means of storytelling. This implies – that telling the right story in the conceptual model – will translate to a positive user experience. The model (or story) should play out with each event – how the user would expect it to in their story that they are creating by using the device. By hitting a key on my keyboard, I expect the letter to appear on the screen. If it did not, my event did not lead to the expected event. There would be a disjoint in my story, from the designer’s. This has big implications in breaking flow and causing frustration for users.

**Blaming the Wrong Thing**

Since, we as users are always writing stories for ourselves, we are in essence trying to match all of our events to causes. So, a lack of feedback for an action would imply to a user that they did something wrong. This might cause them to repeat the action, or worse assume they don’t know how to use your design and leave. This is a user blaming themselves for your design. So, when designing a model, the goal should be to always have feedback for the user – for any action they might take, to help them in matching their perception of the model – to the designed model. Having the user blame the wrong thing (themselves) only leaves them with negative feelings about your design in the reflective stage.

**Learned Helplessness and Positive Psychology**

Learned helplessness as described by Norton is the idea that a user when not given proper feedback, may repeat a task for an arbitrary amount of times until deciding that the task does not work. This learned helplessness can leave a user feeling inept, and incapable of using the design – blaming themselves and leaving your model with negative emotions.

Positive Psychology is not thinking of failures as failures, but as learning experiences. The key takeaways from this is to not blame users as a designer and take difficulties of the users as an area where the product can be improved. Learn from the mistakes of others, as when designing a model – the key idea is to make the model easy to use (minimizing mistakes) for the most amount of users. Turn errors into help messages, allow corrections in software, and assume partial correctness. As to not fully discredit a user in their decision.

Between these ideas, a designer should learn from the mistakes of users, help their users, and minimize areas with insufficient feedback as to lead to repeated mistakes.

**How Technology Users Can Be Falsely Blaming Themselves**

**Norton claims that users of technology often blame themselves for not being able to carry out tasks with devices in many cases, especially when the task seems simple. Regardless of how many other people may make the same mistake with the interface, the blame is often casted on the user of the technology and not the designer. Norton attributes the emotions felt by those affected as “stupidity” or “clumsiness” when in fact it may not even be the user’s fault. This is something that as a designer of a system, you would want to avoid. A simple solution is to put measures in to place to make sure to guide the user through every step, no matter how trivial it seems to the designer. The most important thing when designing technology – is to make it pleasurable for the user. And any user leaving your interface design with feelings of stupidity or clumsiness are certainly not going to remember your device fondly in the reflective part of analysis.**

**Example**

**The Seven Stages of Actions**

1. **Goal (form the goal)**

The objective in mind when performing a task or using a device/software.

1. **Plan (the action)**

The idea in the user’s mind of how they are going to achieve a goal out of the many possible routes able to be taken.

1. **Specify (an action sequence)**

Turning the user’s plan into a series of steps that can be executed (an action sequence).

1. **Perform (the action sequence)**

Taking the specified steps and actually acting upon them.

1. **Perceive (the state of the world)**

The user comprehending what happened when an action sequence has been performed.

1. **Interpret (the perception)**

Making sense of the perceived outcome in the user’s mind.

1. **Compare (the outcome with the goal)**

Objectively looking at the outcome and deciding if it was part of the planned action sequence or not.

The seven stages of actions are largely unconscious. And keeping the action stages unconscious is exactly what an interaction designer aims to achieve. If your product affords interactions that are so seamless and familiar, that the user does not have to use cognitive effort in drawing out all 7 steps – they will have a pleasant or at worst, not memorable experience. Both outcomes far outweigh the concept of having an experience that is memorably bad.

So, the seven provide a model to how user’s think, and designing interactions to conform to this model – that is, keeping the model as subconscious as it is when performing a familiar or pleasurable activity is a step in the right direction to designing pleasurable experiences. This is where Norton refers to keeping users in a “flow” state. By creating seamless transitions between the experiences – you do not break flow, and do not cause unnecessary thought or discomfort for a user. Leaving them with an impression that your design worked and fit their use.

**Discussion and conclusion**

**References**

[1] Norman, Donald A. *The design of everyday things: Revised and expanded edition.* Basic books, 2013.

[2] “The Design of Everyday Things.” *Wikipedia.* <https://en.wikipedia.org/wiki/The_Design_of_Everyday_Things>