Desigining from Both Sides of the Screen: 2章

地引昌弘

2004.9.6

1　2 章 押し付けはしない: 体の動きに目を向ける

□ マウスのクリックやボタンのプッシュなどは、肉体的には大したことではないが、それが本章のテーマというわけではない。しかし、ユーザにこのような負担をどのくらいかけているか考慮することは、重要である。

□ クリックというものは、その処理を他のステップに飛躍させるものであり、利用者からみて快適なものではない。しかしながら、現在の技術は、非常に多くのクリックを課している。

□ クリックを神聖なものとして扱うべきである。

2　「Treat Clicks as Sacred」

□ 本書では、処理のショートカットが生じる操作を総称して "クリック" と呼ぶことにする。

□ Palm PDA の例（製品管理者のインタビュー）

□ Palm のデザインは、引き出しやファイルキャビネットを備えたオフィスの机を参考にしている。

- 机の上にあるものはよく使うもので、あまり使わないものは引出しの中にある。

□ もし、よく使うものが引出しにしまってあれば、その余分な（1 ステップの）操作の累積は我慢ならないものになるであろう。

- よく使わない機能についてはそうではない。

□ 作文の例

- 同じ内容を意味する文でも、少ない単語で構成できれば、そちらの方が強力
- ダラダラ作るのは簡単（絞ることが難しい）

□ UI の設計は文章の編集に似ている。

- スクリーンから一つ一つ剥ぎ取っていく。
- 本質的なものだけを残して、余分なものを剥ぎ取れば剥ぎ取るほど、優れたデザインとなる。

□ Palm のアドレス帳

- 新しく利用するにあたり、何かを起動する or アプリケーションをロードすることがない。
- メインスクリーン上で New ボタンを押すだけでよい。
- 既存のアドレス帳を使いたい場合は、単にそれに触れるだけでよい。

□ 名前の検索もクリックは 2 回だけ。

- 一つは Palm の電源を入れ、もう一つは該当する名前に触れる。
- 名前の最初の文字を入れると、該当する名前を持つ先頭の人物までスクロールする。
- "Look Up" フィールドに触れる必要がない。
- 文字の入力を受け付けるフィールドも一つだけ。
- 検索結果には、名前だけではなく電話番号も表示される。検索結果の名前から、さらに電話番号を検索するといった作業がない。

□ 大半の PC におけるアドレス帳では、はるかにクリック数が多い。

- 典型的な PC ベースのアドレス帳の例
- 7 対 2

□ Ultimate TV の例

- ユーザを自身が記録したリストに導くボタンがある。
- TiVo では記録したリストを見るためにクリックが 2 回必要

□ さらに、全てのクリックが同じようなものだと判断してはならない。

- ボタンのクリックやチェックボックスは、狙いを付けてクリックする。

- PC や Web のメニュー選択では、これに加えてマウスのボタンを押しつつ移動させ、その後離すという操作が必要
- これはさらなる身体的負担を課しているため、クリック 2 回と換算してよいかも知れない。
- サブメニューは、さらに長くマウスを保持しながら、その動きを制御する必要があるので、もっと悪い。クリック 3 回と考えるべきである。
- スクロールさせ、正しい位置で離す動作は、クリック 1.5 回と考えるべき。
- ドラッグ & ドロップは 2 回さらに、ソースの中でクリックし、直後に正しい宛先まで持って行くのも負担
- キーボードとマウスの結合は、二つの手の調整が必要なので、クリック 1.5 回

□ これらの操作は、運動としてはさして困難とは言えないが、努力を強いるということに留意すべき。

- ノートパソコンにはマウスが付いていない場合がある。
- トラックボールやタッチパッド、ジョイスティックでは、クリックしながら動かすことが煩わしい。
- ハンディのある人を想定すると、もっと大変
- 移動デバイスも、クリックは大変

3 「実例」

□ デルタ航空の例

- 存在しないフライトを入力すると、そのフライトが存在しない旨と New Search ボタンが表示される。
- New Search ボタンを押すと、フライト入力画面に戻ってしまうため、余分なクリックや労力が課される。
- 該当するフライトが存在しないことだけを表示して、入力は継続させるべきである。

□ バーンズ & ノーブルの例

- 具体的なイメージがよくわからなかった (single match?)。
- B&N では、検索結果が表示された後、さらにクリックが必要だが、アマゾンや IMDB では、該当する情報が同時に表示される?

□ ゼロックスの例

- ボタンだけではなく、タッチパネルを備えている。
- コピー中に別のコピーを指示すると、仕様中を示すウィンドウがタッチパネルにポップされる。
- 前のコピーが終了してもポップアップは表示され続け、それにタッチしても何も起こらない。
- タッチパネル内の別の場所に触れ、次にボタンを押す必要がある。
- もっとショボイコピー機でも、要求はキューイングされるのに。。

□ シボレーの例

- クリックは神聖なもの: クリックなし。
- 自動車に近付いただけでロックが解除されるし、離れると自動的にロックされ、それを教えてくれる。
- キーを出す必要もない。
- 執事と同じ。

□ Quicken, Netscape, Eudora の例

- "name suggestion" 機能が良い。
- 例えば、URL を入力すると、入力した分にマッチする過去に訪れたアドレスを補完してくれる。
- 該当する名前が表示された段階で、URL の入力を止められる。
- 全く新しい URL の入力でも、副作用を引き起こさない。
- ブックマークより良い。
- Quicken では、メールの宛先を入力するだけで関連したエントリーも表示される (Tab で追加していける)。

□ パシフィックベルの例

- 顧客サービス番号に電話すると、自分の電話番号を入力するように指示される。
- オペレータが出ると、もう 1 度自分の電話番号を言わされる。
- つまり、システムはオペレータへ必要な情報を渡していない。システムは、ある処理の自動化だけ。

□ これらの事例では、プログラミングの努力が足りない。

- ユーザの負担に考えが及んでいない (月並な言い方ではあるけれど)。

4 ”Do You Really Mean It?” Pop-Ups

□ クリックは神聖なものであるというルールに最も違反する振舞いは、「実際に行ないますか」と呼び掛けることである。

□ このうるさいポップアップは、様々なパターンであらゆる場所に出現する。

□ このポップアップが一つ現れるたびに、ユーザの思考の流れを阻害している。

- ユーザはコンピュータに指示を出したのに、コンピュータから質問が来るし、クリックさせられるし、ことによったら異なる入力装置に手の位置も変えさせられる。

□ エンジニアは、誤りからユーザを保護するためにこのポップアップを表示する。

- このポップアップを表示しないと、ユーザは回復不能な間違いをするかも知れない。

□ しかしながら、このポップアップは、しばしば必要でないときに出現する。

- 多くのユーザは、何も考えずに Yes と答えてしまう。

□ 「実際に行ないますか」ポップアップは、エンジニアがユーザに責任を転嫁する方法

- 「ユーザに確認させた」

□ 「実際に行ないますか」ポップアップが必要な場合は、

- 1. ユーザが多くのデータを失う可能性がある時
- 2. データの回復に、多大な労力が必要な時

□ 本来、このような状況は滅多に起こってはならない。

- バックスペースを使って文字を消す時を考えてみると。。

□ 「実際に行ないますか」ポップアップは Undo 機能を実装することで回避できる。

- しかし、ユーザにはまだ確認の依頼が来る。
- 例えば Windows のゴミ箱ゴミ箱に入れるたびに、確認を求められる。
- Photoshop も同じ（具体的にはよくわからないけど)。
- Eudora も同じ。メールの作成を中断すると、セーブしますかと聞かれる。
- mailto リンクも同じ。どのページにも付いているので、間違えてクリックしてしまうことがある。間違えたメールツールを閉じようとすると、保存するかどうか聞かれる。十分に思考の流れを壊している。

□ 自動保存機能の採用でも「実際に行ないますか」ポップアップを回避できる。

- 保存したかどうかを気にするだけではなく、プログラムを終了する時も質問に煩わされることがない。
- 利用者は作業に集中できる。
- 実装も容易

□ 自動保存機能は余分なものまで保存してしまうという懸念は正しい意見である。

- しかし、保存しないで終了したことによる後戻りより、保存を怠ったことによるデータの損失の方が痛手となる。

□ これはデザイナとエンジニアとの競合点とも言える。

- エンジニアは、まれなケースでもプログラムが正しく動作するように訓練される。
- まれなケースは多々あり、配慮を怠るとシステムが壊れてしまうため、エンジニアはこれらに多大なエネルギーを投入する。
- デザイナは、後戻りの方をより重く考えるので、共通のケースに磨きをかける方に時間を費す。

□ インターフェイスの設計においては、共通なケースをエレガントに、まれなケースは適切に扱われているかを確認しましょう。

- デザイナも、共通であろうとまれであろうと必要な処理はきちんと伝える必要があることに注意しましょう。

□ Eudora の例

- 特に指示せずにメールの作成を終了しても、Out Folder を開くことで作成を再開できる。

□ なぜクリックが必要かを考える代わりに、あなたが何をさせようとしているかを考え、ユーザが悩むことなく実行できる方法を見つけ出しましょう。

□ 「実際に行ないますか」ポップアップは無礼である。

□ 以下では、クリックを減らせる幾つかの方法を示す。

5 「Remember Where They Put Things」

□ 多くのアプリケーションは、UI をカスタマイズする機能（preference）を備えている。しかし、往々にしてユーザはその機能を使わずに、自分達独自の方法でカスタマイズする。

□ デスクトップアプリケーションでは、ユーザがウィンドウの場所やサイズを常に変更するが、一度ウィンドウを閉じてしまうと、ユーザの選択（preference）は忘れられる。

□ 次にそのウィンドウを開く時は、またカスタマイズし直さなければならない。

- タブの選択/スクロールなど全て同様

□ ウェブブラウザは、どこまでスクロールされたかを覚えているので良い。

- ページの最後まで見てから別のリンクをクリックし、再びそこに戻って来ても、ページの先頭へは飛ばされない。

□ TiVo（Personal Video Recorder）も良い例である。

- TiVo のインターフェイスはメニュー形式であるが、テレビを消した後も、ユーザがスクロールした場所や選択項目を覚えており、次にテレビを点けた後は、前回のメニューが示される。
- ユーザがいる場所を思い出させることにより、タスクの継続に必要な労力を軽減させる。

□ ウェブのタブは、そのように機能しない。

- 800.com の例
- Electronics タブで Video/TV の項目を選択してから Music タブへ移り、再度 Electronics タブへ戻ると、Video/TV ではなく、Electronics タブの先頭に戻される。
- ユーザは、買物をする前に色々と見てまわりたいが、線形のパス上で買物ができない。

□ 幾つかのウェブサイトは、広告のためにユーザへわざとクリックさせるものもある。

□ データベースへのアクセスなど、ユーザセッションの状態をもとに戻すための実装コストが高いアプリケーションも存在するが、大半の PC アプリケーションでは、最後のタブを覚えておくことぐらい難しくないはず。

□ ユーザに他のウィンドウを開かせ、もとに戻った時に以前の場所ではない所へ戻すことは無礼な振舞いでもある。

□ ユーザの位置を保存しておくことは、携帯電話やカメラといった入力メカニズムに制限のある装置では特に重要

- メニューの選択を循環させる必要がある。
- 一度あるメニューを選択し、次に他のメニューを選ぼうとすると 5，6 レベルは移動することがざらにある（図 2.8）。
- このような制限のあるインターフェイスでは、TiVo のようにユーザがどこでスクリーンを消したかを覚えておくことが重要になる。

□ ドキュメントの保存でも同様な事情がある。

- Excel の例
- 新しいスプレッドシートを作成し、それを保存しようとすると、毎回デフォルトのフォルダを提示される。
- Photoshop も同じ（詳細はよく分からないけど）。

□ 多くのアプリケーションでは、デフォルトの位置を変えられるようになっているが、ほとんどのユーザはその機能を利用しない。デフォルトとは異なるフォルダに保存した場合、（その場所を示す?） オプションは有益と言える。

□ ユーザが何時何を決定したかを次回に思い出させることはより有効である。

6 「Remember What They Told You」

□ ユーザに個人情報を提供させる場合は考慮が必要以前に答えた質問を再度問いかけることがどのくらい厄介か。

- それは無礼でもある（信頼を失うことにもなる）。
- ウェブサイトに多く見られる。
- アマゾンの例

□ 携帯電話は良い例と言える。

- 一度登録した相手からかかって来た場合は、電話場号ではなく名前が表示される。
- 未登録の者からかかって来た場合でも、容易にアドレス帳へ登録できる。

□ 「Remember What They Told You」と「Remember Where They Put Things」は、エンジニアにはしばしば見落とされる。

- 使いやすさよりコア機能拡充に力が注がれる。

□ これらの機能のアーキテクチャへの組み込みが早ければ早いほど、新しい機能拡張にも対応しやすくなる。

7 「Stick with a Mode」

□ 手の位置を変更することが、破壊的な所業あることは既に述べた。

- キーボードからマウスへの移動は特に厄介
- 手を戻すために常に見ていなければならない。

□ モードの変更を回避する良い方法は、入力方法を多数提供すること。

- 大半のデスクトップアプリケーションは、マウスだけではなくボタンでも入力できる。
- Internet Explorer の例（ちょっとピント来ないが)
- Netscape はボタンを使えない?

□ もう一つの方法は、自動的に入力フィールドへポインタが移ること。

- マウスを掴む必要がない。
- 主な検索エンジンの中で Google.com だけがそうしている（本当?)

□ これ以外に、ユーザに意味のある順番でタブを付けることを許すという方法もある。

- ユーザがこのようなカスタマイズをするのか、ちょっとピント来ない。

□ 標準のキーボードやマウスを備えない装置は、さらに注意深く設計するべき。

- Psion PDA の例
- キーボードと尖筆の両方を備えているが、非常に使い勝手が悪い。

□ 携帯電話では、番号の入力と電話の利用で操作が切り替わる。

- AT&T のボイスメールシステムの例
- キーを押す必要がない。ボイスメールを呼び出すと、メッセージが何件あるかを知らせて再生を始める。

□ この章で述べたかったことは、ユーザの物理的な労力について考慮し、ユーザに問いかけるという動作を減らすこと。