情報システム技術論'02 ＃4

久野　靖 *

2002.6.13

1　はじめに

□ 今回の内容…

- 3 章「サイトデザイン」
- デザイナはどういう風に Web を作っているのか?

2　サイトデザイン

□ とある調査

- 「簡単な問題の回答をホームページで探す」→できた人 42%
- 「Web サイトで」かと思ったが「ホームページで」みたいだ!
- 「求人広告を探して申し込む」→ 26%
- もちろん、ダメなサイトを特に用いたわけではない
- 50% × 50% = 25% → 1 個所でもダメなら全体はもっとだめ

□ もし、ホームページだけで済まない作業だと→さらにずっと悪い!

- WWW は文書を読むための環境としてデザインされている
- 何らかの作業を行なうための環境ではもともとない
- (「何らかの作業を行なうための環境」だったらどうしたと思うか?)

□ ゆえに、「何かをうまく行なってもらう」には→「シンプル」を何よりも優先する必要がある

□ よくない例…

- 「工事中です」のページ (p143)
- 「?」が検索機能を表しているページ (p143)

2.1　ホームページ

□ 「ホームページ」って何ですか?

□ (1) ブラウザの「ホーム」ボタンで行くページ

□ (2) Web サイト/Web プレゼンテーションの起点のページ ←☆

□ 機能その 1…

- 「ここはどこ?」「このサイトは何?」に答える
- これらがはっきり分かるように→大きなロゴ、目的の説明
- はじめてのユーザがターゲット

□ 機能その 2…

- ナビゲーションの起点
- 階層構造のトップ…とは限らないかも
- よく訪れる人のための機能を入れ込んでしまう (例: 航空会社→予約機能)
- ニュースや宣伝→まあ分かるがあまり多くしないのが吉
- 例外:　ニュースサイトのホーム (しかしそれでも、特定事項の記事を探しに来る人は多い)

□ ユーザは検索が好き、リンクを辿るのは嫌い

- 従って検索機能は必須 (そうなの?)

□ まとめると…(1) サイトの内容、(2) ニュース、(3) 検索、が 3 大機能

- ダメな例…p146〜p150

2.2　ページの幅はどうする?

□ 「幅を決めてはいけない」(イントラネットを除く)

- どうしてもやりたければ 600 ピクセル以下に
- 1995〜1997 年の調査→ 525〜598 ピクセルが平均であった
- 「liquid デザイン」→幅がいくつでもいいように作るのがよい

*筑波大学大学院経営システム科学専攻

2.3 スプラッシュページ

□ …「Enter」とかだけ書いてあるページ…ナゼ?

□ スプラッシュページを作る理由

- ウェルカムメッセージを見せる
- ナビゲーションなどの要素なしにロゴなどを見せる
- 一定時間待っていると「自動的にホームへ移動」

□ そんなものは無駄だから作らない方がよい

- 最初のページから直接有効なページであるべき

2.4 ホームページと内部ページ

□ ホームページではサイト名などを大きく示す

- では他のページではサイト名は要らないか?

□ 他のページでもよそからそこへ到達することはある

- 従って「何のサイトか」は分かるようにするべき
- しかし大きく表示する必要はない→本来の内容を重視

□ ホームページへのリンクはどのみちすべてのページに入れる (ホームページそのものには入れない!)

- ロゴマークも入れるのだったら、それをホームへのボタンにするといいかも
- しかし気が付いてもらえないかも→ボタンやリンクはやはり要る

□ 有名でないサイトほど「ここは何」という情報を少し増やす

2.5 深層リンク

□ 「直リン禁止」はよくない…前に出た

- 入口は多数あってよい
- ホームページを通ることを強制→ユーザにそのサイト特有の規則を学習することを強制していることになる
- affiliate program

2.6 メタファ

□ ウェブデザインではメタファの使用が多い

- 見た目楽しい、ひきつける
- 一方で本来の目的 (ユーザに使いやすい) を置き去りにする危険
- メタファに合わせてサイトを作るのはいまいち…

□ メタファの利点…

- デザインの統一感
- ユーザの理解を助ける

□ 例: e コマースでの「ショッピングカート」

- カートに入れた商品はまだ購入はしていない→スーパーと同じ
- しかし完全に同じというわけではない…

□ 完全に同じでないところが色々ある

- 5 個買うのに、5 買い取り出してカートに入れる必要はない
- キャンセルするのに元の棚に戻す必要はない
- これらの違いが混乱やトラブルをもたらすことがある

□ 意味のないメタファの例 (p158〜p161)

□ 「ショッピングカート」は広く使われているのでもはやメタファというよりインタフェース標準

- その標準的な動作に一致している限りにおいて、ユーザは何も新たに学習しなくて済む→大きな利点
- 逆に、「ショッピングカート」が適していなくても標準として確立しているためそれに従わざるを得ないケースも

□ 「Shoping sledge」(ウインタースポーツのサイト) →ユーザは???

2.7 ナビゲーション

□ Web は広大な空間→「単にクリックするとどっかへ行く」ではうまく活用することが難しい

□ 3 つの質問に答えられる必要

- ここはどこ?
- 今までどこにいたの?
- ここからどこへ行くの?

2.8 ここはどこ?

□ 最も重要な質問 (これが分からないと他の質問も無意味)

□ 2 通りの形で答えを示す必要

- WWW 全体の中でどこにいるのか→各ページのロゴ等
- サイト内のどこにいるのか→色々な工夫

□ 普通のサイトと大幅に違うことをすると評判が悪い

- ユーザの予想を裏切ってはいけない
- 極端に変わったインタフェースはよくない

□ WWW 中のどこ→全ページにロゴマークを

- いつも同じ位置に表示されるべき→左上隅がよい

□ サイト内のどこ→

- マップを常に表示し、その中で「あなたは今ここ」をハイライト
- 階層構造サイトなら「ルートからここまでの経路」を表示
- アイコンバーなどで「今いる位置」が分かるようにする

2.9 今までどこにいた?

□ どこから来たかは単純な HTML ページでは表示しようがない

- ブラウザの「戻る」ボタン/履歴リストは有効な機能
- 訪問したリンクは色が変わるのも有効 (標準の色のままにするのがよい)
- 同じページに何回も行ってしまうことを防ぐのはとても有効

□ ブラウザでもっと 2 次元的な「地図」を作ってくれるといいと思う…

- 中村さんの研究?

2.10 これからどこへ行く?

□ 見えているリンク、見えていない行き先まで含めて提示できると有効

□ ハイパーリンクの分類

- 埋め込みリンク→文章の中に埋め込まれている
- 構造リンク→サイト構造に従ったリンク。「上」「次」より具体的な内容つきにした方がよい
- 連想リンク→これも関係ありますよ、という情報

□ スタイルは「青い下線つき」という標準に従うのがベスト

- ボタンやメニューなどはよく見ないと分からないし、訪問しても色が変わってくれないという問題がある。

2.11 サイト構造

□ 情報アーキテクチャ(information architecture) → コンテンツとその提示構造の両方からの構造 (設計) → 使いやすさのために本質的

□ ほとんどのサイトは階層構造→単に階層になっていればいいというものではない→どういうことが問題?

- 直線構造も部分的にはあってよい

□ 構造を設計すること (成行きまかせが実はとっても多い)

□ 組織図をそのまま構造にしないこと

- 責任分担は楽だろうが、ユーザはあなたの企業がどんな部署に分かれているかは知りたくない

□ ユーザの視点からの分類・階層構造が重要「製品」「人材」「投資」など

- 各ページで階層の各段階の分類がすべて表示されていると現在位置が分かりやすい
- 各階層での他の (隣の) 選択肢まで見られるとなおすばらしい

2.12 階層の幅と深さ

□ 幅優先デザイン→例: news.com

- 全ページ左側にトップレベル選択肢のリストが並んでいる
- このために 20%もの領域を全ページで割く価値はあるのか?
- ロゴマークとしての役割は確かにある

□ 深さ優先デザイン→例: useti.com

- breadcrumbs(パンくず) →ホームからここまでの経路を常に表示
- 階層関係がきちんとしていないと効果がない

□ 幅と深さと両方入れることもできる→スペースを取る

- 階層メニューのように DHTML を利用する方がよさそう

□ Fisheye View →現在いる付近だけクローズアップして詳しく

□ Sun MicroSystems の例→ L 字型→上のバーでトップの選択肢、左側でその中の選択肢を表示→巨大だが構造が整理されたサイトむけ

2.13 ユーザコントロール

□ 従来のインタフェース→デザイナがユーザの動きをコントロール

□ WWW →どのサイトも一瞬しかいない→学習してもらうことは無理→ユーザがコントロールするように→標準的なスタイルに従う

2.14 情報の山の管理

□ GUI → Alto や Mac の文化→システム内のドキュメント数はごく少なかった

□ WWW → 10 億ページ、数億のユーザ→現在のブラウザがよいかは?

□ もっとソフトの機能によるサポートが必要では (例: 動的サイトマップ)

- サイトマップ情報を XML で配付してブラウザがマップを表示するとか…

2.15 ナビゲーションを散らかさない方法

□ 集合化→複数のものを 1 つにまとめる (サイト自体がそう)

□ 要約→縮小版や抜粋を見せる

□ 絞り込み→あまり価値がないものは削る

□ 切り取り→先頭だけ見せる等→詳細はリンクをたどると見える

□ 例→サンプルだけ見せて「残り○○個」

2.16 サブサイト

□ 巨大なサイト→それを 1 つの構造で統一するのは難しい

□ まとまった部分構造→サブサイトとして分離

- 親サイト側からはホームページで代表させられる
- サブサイト内ではそれ単独で自立した構造を持ってよい
- スタイルやナビゲーションの仕組みなどは親サイトと共有できる

2.17 検索機能

□ 調査→ユーザの 50%は検索優先、20%はリンク優先、残り 30% は混合型

- 検索する人が多いからといって、構造がいい加減でいいわけはない
- 検索で見つけた後動き回るのにも構造が重要
- サブサイト限定の検索はあまりよくない (サイトの構造を分かっていないユーザが多い) ←どうかな?

2.18 高度な検索

□ 「あなたは犬と猫を飼っています。あなたのペットたちに関する情報を検索するには?」→「cats AND dogs」→失敗

- 論理式を用いた検索はしない方がよい
- 簡単な検索機能とは別に用意するのならそれでもよい

2.19 検索結果の表示

□ 確率の高いページから順に表示する (ユーザは上から見て行く)

□ 同じページが複数回出てこないようにする

- よい検索結果を出すのは工夫必要
- 例: Google のページランク/グループ化

2.20 Meta タグによる指定

□ 要約 →<meta name="description" content="要約文">

□ 検索語→<meta name="keywords" content="語 …">

- 人間が指定したものの方が自動抽出よりもずっとよいから

2.21 キーワード

□ 従来の情報検索と比べて、WWW ではユーザが入れる語数が多くなっている

- WWW は情報量が巨大なので、語数を多くして絞る必要

□ 検索語の入力欄は十分幅広くした方がよい→語数を多く入れてくれる→精度が高まる

□ 結果の提示時に「検索語があった個所」を強調表示しつつその周辺を見せてくれる→実際に行ってみなくても判断できる

2.22 URL のデザイン

□ 「/」を使ったのはよくなかった…Tim

□ 「http://」は省略されてしまうことが多い (USA では。いいのか?)

□ 「www.company.com」と「company.com」と両方用意した方がよい

□ 分かりやすいドメイン名を押えることはとても重要

□ ユーザは URL を頼りにナビゲーションする

- URL の先はどうなっているか示せるといいのだが…
- 意味のある名前を各要素に使うのがよい
- 途中から先を削除すると上の階層のページになるのがよい

□ URL の規則

- できるだけ短くする。短いほど誤りが起きにくい
- 一般的で自然な言葉を使う。知っている言葉は覚えやすい
- すべて小文字に。混在すると記憶が大変。
- 特殊文字は避ける。区切りはハイフンだけ使う等。
- 「o」と「0」、「l」と「1」

2.23 URL の保存

□ URL が腐ってたどれなくなってしまうのは最低

- 内容を変更したければ URL は同じで内容だけ差し替える
- 定期更新コンテンツは恒久版 URL と対で広める

2.24 URL の広報

□ すべての製品にその製品紹介ページの URL を入れるべし

- パッケージに印刷で入れるのは簡単

□ 古い URL もサポートはしないといけない

- 古い URL のアクセスに対して自動的に別ページに誘導できる (サーバの機能)

2.25 掲示板は…?

□ ユーザとスタッフが議論できることがよいとする意見も…

- 巨大サイトではとても全部に対応しきれない、大変すぎ
- ユーザどうしで議論してもらったら? →喧嘩とか荒れて大変

2.26 まとめ

□ サイトデザインの問題の多くは技術の進歩で解決が可能

- 現在のところはまだ実現していないだけ
- (本当にそうなのかどうなのか、久野は疑問)

□ 問題は「ユーザビリティが人の手を借りていること」←本は誤訳?

- すべてのデザイナーとライターが協力する必要がある
- どんなに検索が進歩しても内容がヘタレだったらどうにもならない

□ 会社のニーズではなくユーザのニーズを指針とするべき

- よい (ユーザ指向の) インフォメーションアーキテクチャ
- よいリンクのつけ方
- よい見出しのつけ方

□ 今日は「悪いサイト」だらけ→「良いサイト」が増えてユーザが悪いサイトを拒否するようになって欲しい

3 各自のサイトデザイン発表

4 実践編

□ だいたいテキストの指示どおりにナビゲーションデザインしたプレゼンテーションを作ってみた→スタイル指定してみる

□ 演習方法:

- 「cd」でホームディレクトリに行く
- 「tar xvpf /u1a/kuno/work/ist03.tar」を実行
- 各自のディレクトリ WWW の下に「lang」ができるので、この下のファイル「std.css」を色々手直ししてみる
- 「http://w3in/ ユーザ/lang/」で見られる

4.1 準備 1

□ まずはページ群をひととおり見てください

- どのような構成になっているか分かりますか?
- ナビゲーションのやり方は分かりましたか?
- どのような問題があると思いますか?

4.2 準備 2

□ まず各ページが大きく 3 つに分かれていることを確認→ std.css に次の 3 行を入れる

```
div.main { border: solid black 2px }
div.navmenu { border: solid blue 2px }
div.navbar { border: solid red 2px }
```

□ 本体のマージンが邪魔なので 0 にする

```
body { margin: 0px }
```

□ これを見て本体をどこに置き残り 2 要素をどこに置くか決める

4.3 各要素の位置ぎめ

□ 本体の上マージン、左右マージンを決める

```
div.main { margin-top: 5ex; margin-left: 20% }
```

□ ナビメニューを左側に縦に詰める

```
div.navbar { position: absolute; top: 0px }
div.navbar { width: 20%; height: 100%; padding-top: 6ex }
```

□ ナビバーを上に詰める

```
div.navbar { position: absolute; top: 0px }
div.navbar { width: 100%; height: 4.8ex }
```

4.4 色の調整

□ 各領域の色を調整する

```
body { background-color: 色 } ←既にあるのを修正
div.navmenu { background-color: 色 }
div.navbar { background-color: 色 }
```

4.5 配置の調整と記号

□ ナビメニューの並びを縦に変えて記号を入れる

```
span.menu1, span.menu { display: block;
                        position: relative; left: 1ex }
span.menu1:before { content: "☆" }
span.menu:before { content: "→" }
```

□ ナビバーの中も同様に

```
div.navbar { padding-left: 2ex }
span.navbtn { position: relative; top: 1ex }
span.navbtn { border: green outset 6px; padding: 2px }
span.navbtn:befer { content: ">>" }
```

4.6 仕上げ…

□ 位置ぎめや詰め物などを調整

□ ここまでにやったところも気に入るまで直す

□ 枠を表示させている 3 行を「/* ... */」で囲んでコメントアウト

□ その他気に入るまで調整…

5 次回までの課題

□ ニールセン本は今回までにして、最終回は別の論文や本から取り上げようと思います。次の 3 つの論文を配付しますので、(1) は必ず読んで来てください。(2) は 50%くらいの濃さで読んで来てください (カンタンですからすぐ読めるはず)。(3) は興味があれば読んで来てください。

- (1) Mark W. Neman, James A. Landay, Sitemaps, Storyboards, and Specifications: A Sketch of Web Site Design Practice, Proc. Designing Interactive Systems 2000, pp. 263-274, 2000.
- (2) Mark W. Neman, Jason I. Hong, James A. Landay, DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design, Proc. CHI 2000, pp. 510-517, 2000.
- (3) James Lin, Michael Thomasen, James A. Landay, A Visual Language for Sketching Large and Complex Interactive Design, Proc. CHI 2002, pp. 307-314, 2002.

□ 各自の「10 ページ程度の Web プレゼンテーション」は次回完成をめざして作成してください。まあ間に合わないかも知れませんが、できているところまでで発表していただきます。

□ レポート予告: 「10 ページ程度の Web プレゼンテーションを作成し、その内容を報告せよ。またそこにおいて本科目で学んだことやその他 Web ユーザビリティに関する事柄をどのように活用しているかについても論ぜよ」→〆切は 7/15(月) 一杯とします。