人にやさしく、ときどき厳しいシステム
A system as smooth as silk, with a few tight knots.

森田真基
Masamoto Morita
株式会社 朝日ネット クラウドサービス部
ASAHI Net, Inc.　Cloud Service Department

要旨
ソフトウェアやウェブサービスに対して「使いやすさ」が求められているのは疑いがない。特に、専門家以外のユーザや、多様な年齢層のユーザが使用するシステムにおいては「使いやすさ」は利用率にも直結する。
「使いやすさ」に影響を与えるものとしては、システムの仕様やＵＩ（ユーザインターフェース）など、あらゆる要素が関わるが、重要なのはそれらを設計するにあたり、ユーザたる「人」の特性をどのように想定するかである。
本稿では、多くの教育機関で採用されているクラウド型学習・コミュニケーションサービス[1] を設計・提供した経験をもとに、「使いやすさ」の意味を見直し、「ユーザのモデル」を定義する。そしてそのモデルを元に「使いやすい」システム設計の際に注意するべき点を考察する。

1.　「使いやすさ」が求められるシステムとは

本稿で述べる「システム」とは、人が直接操作する情報システムである。たとえば、ウェブサービスであり、ソフトウェアであり、携帯電話やケーブルテレビなどの操作画面すべてを含む。

さて、「使いやすさ」がどの程度求められるかは、そのシステムの機能の数と、ターゲットとするユーザの種類による。図１は、横軸にシステムの機能数、縦軸にユーザの種類を置いたものである。「使いやすさ」が特に求められるシステムとは「○」の領域に属するもので、本稿で扱うシステムもこれに限定する。

そもそも機能が少ない場合、その時点で「使いやすい」場合が多い。機能が増えれば増えるほど、使いやすさ」が求められてくる。一方、そのシステムを使うユーザが特定の専門家に限られる場合は、たとえ使いにくくとも問題がない。しかし、不特定多数の専門知識がないユーザをターゲットとするならば、機能がある程度多くなった時点で「使いやすさ」は求められる。

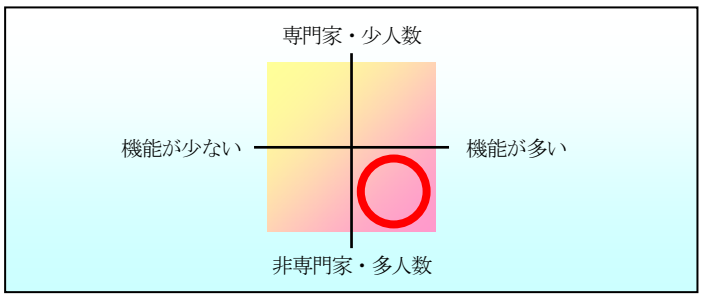

図 1　「使いやすさ」が求められるシステムの領域

2.　「最悪のシステム」

まずは、最も「使いにくい」システムを挙げる。一つは、［不具合］を含むシステムである。そもそも操作することが不可能な、プログラムあるいは仕様に致命的な欠陥を含んでいるものを示す。このようなシステムは、どのように優れた設計であったとても、即「使いにくい」と判断されてしまう。

同様に、設計がどんなに優れていても「使いにくい」と判断される要因は速度である。たとえば画面の切り替わりや、処理の速度が、ユーザの主観的に耐えられないほど［遅い］時点で、そのシステムは「使いにくい」と判断されてしまう。

［不具合］を含む、もしくは［遅い］システムは、決して「使いやすい」とはいえない。その評価以前の「最悪のシステム」となる。

3. 「使いやすさ」の定義と、３つのレベル

本稿では、「使いやすさ」とは、「使いこなすことが簡単であるか」と定義する。どのようなシステムであれ、ユーザは使い方を学習する必要がある。この学習がスムーズにできるか否かが、「使いこなすことが簡単であるか」に影響を与え、ひいては「使いやすいか」に繋がる。

もちろん、システムの使い方を、１００％習得したユーザ（＝使いこなしているユーザ）にとっても、システムに対する不満があったり、満足したりしているだろう。しかし、本稿ではこの「満足度」は「使いやすさ」から除外している。

システムは「使いやすさ」の観点で、次の３つのレベルに分類される。

[LEVEL 1] ― 学習に負荷を感じるシステム

ユーザがシステムの使い方を学習する際に、何らかの「負荷」を感じるシステムである。負荷を感じるゆえに、自ずと学習に対して消極的になる場合が多い。「負荷」については次章でまとめる。この「負荷」の数を減らすことこそが、「使いやすいシステム」を構築する事に直結する。

[LEVEL 1] を端的に表現すれば “Difficult to Use” となる。

[LEVEL 2] ― 学習に負荷をあまり感じないシステム

ユーザが使い方を学習する際に、負荷を感じることが少ないシステムである。この時点でシステムは「使いやすい」と判断される場合が多い。[LEVEL 2] を端的に表現すれば “Easy to Use” となる。

[LEVEL 3] ― 意識せずに学習できるシステム

第３のレベルは、やや特殊である。ユーザが使い方の学習を、意識せずにできるシステムのことである。意識していないので、負荷も感じない。 [LEVEL 3] は、 “Intuitive” （直感的）と言える。「使いやすさ」が求められるシステムであれば、[LEVEL 2] に留まらず [LEVEL 3] まで高めるべきである。

4. ユーザモデルと「負荷」

さて、学習の際に感じる「負荷」とは、何であろうか。新しくモノを覚える、そもそもこれ自体を負荷と感じる人もいる。一方で、楽しいことならば学ぶことを負荷と感じない人もいる。

そこで、負荷を考察するにあたって、負荷を感じる主体であるユーザのモデルを定義してみたい。このモデルは、筆者がこれまでシステムを構築してきた経験で学んだもので、３つの特性からなる。そして、それぞれの特性にあわせて、典型的な「負荷」を挙げてみたい。

4.1. 「使いやすい」システムを作る上でのユーザモデル

ユーザは、

(A) 興味がない事は嫌いだが、(B) 齟齬に関しては敏感 であり、その一方で (C) 学習意欲は高い。

ユーザの特性(A) ：興味がない事は嫌い

人は、興味がないことを我慢して続けられない。また、たとえ興味があることであっても、興味を失うと、とたんに不快に感じてしまう。

ユーザの特性(B) ：齟齬に関しては敏感

人は、齟齬（＝不一致）に対して、敏感である。専門知識がなくても、またそれを言語化できる知識がなくても、直観的に齟齬を察する。齟齬や矛盾を察すると不安や不快を感じる

ユーザの特性(C) ：学習意欲は高い

人は、新しい知識の習得に対しては積極的である。自分で考えて学習する分には、試行錯誤も厭わない。そして習得や発見をしたときに喜びを感じる。ただし、試行した結果が自分の期待と違った場合は、不快に感じてしまう。

4.2. ユーザモデルに対応した「負荷」と、具体例

ユーザモデルは前章で述べたとおり、３つの特性からなるが、それぞれの特性に対応した「負荷」の具体例を挙げてみたい。

負荷(A) ― 過剰な説明

ユーザの特性(A) に対応した負荷を、負荷(A)と呼ぶ。負荷(A)として典型的なものは、膨大なページ数のマニュアルや難解な説明文である。親切丁寧な説明や解説であっても、ユーザの興味が削がれる分量であったり、あるいはユーザが興味を維持できなくなるほど難解であったりした時点で、ユーザにとっては「負荷」となる。 マニュアルは、すべての機能を説明するために長くならざるを得ない場合も多い。しかしそれでも、ユーザにとって量が過ぎれば負荷以外の何物でもない。

負荷(B) ―用語や UI の不一致

ユーザの特性(B) に対応した、負荷(B)として典型的なものとしては、システム内の用語の不一致が挙げられる。用語が、画面ごと、あるいは説明書と画面とで統一されてないシステムというものは、多々ある。このようなシステムにおいて、「この言葉とこの言葉が一致していないからよくわからない」と違いを認識して言語化できるユーザはむしろ少ない。しかし、言語化できなくても、ほとんどのユーザは違和感や不快感を感じてしまう。用語だけでなく、ＵＩの不一致に対してもユーザは敏感である。ある画面で左側にボタンがあり、別の画面で同じ機能のボタンが右側にあれば、不快と感じる。どんなに瑣末な齟齬であってもユーザは直観的に感じ、システムの学習の際に「負荷」と捉える。

負荷(C) ―試行錯誤を制限する仕様と、期待を裏切る反応

ユーザの特性(C) に対応した、負荷(C)はシステムの設計や仕様に関連する。ユーザが自分で考えて行動しようとした際、それができない仕様であるとユーザは「負荷」と感じる。また、その行動ができたとしても、動作結果がユーザの予想と一致しなかった場合に「負荷」と感じてしまう。ユーザの行動や予想は、システム設計者の想定に収まっていない場合もある。しかし、収まっていようがいまいが、ユーザは期待通りのことにならなければ「負荷」と感じてしまう。

5. 各負荷の対策 ― 使いやすいシステムを構築するために

負荷(A) の対策は、携帯電話のマニュアルなどで、ある程度の具体例が見受けられる。機能を詳細に記述した詳細なマニュアルと、「クイックマニュアル」といった、とりあえずシステムを使う為に必要最低限なマニュアルと２つ用意する方法である。また、説明の量を減らすアプローチも重要であるが、マニュアル自体の文体に対してもユーザの興味を引くような表現や、より平易な表現を使うなどの工夫も負荷(A) を減らす上で重要となる。

負荷(B) の対策は、システム内の用語や表現、ルールの統一である。これは、システム全般に関わる用語の「辞書」や、ＵＩの「ルールブック」を策定することで統一ができる。細かな言い回しなど、終始一貫して整合性を保つことで、負荷（B）を減らすことができる。

負荷(A) と負荷(B) を取り除くことに成功すれば、システムの「使いやすさ」は [LEVEL 2] ― 学習に負荷をあまり感じないシステム ― にまでは達することができる。

問題は負荷(C) である。多くのシステム設計者は、ユーザの特性(C) を見逃し「人間は学習に対して消極的」という悲観的な前提に立っている。そのため、過剰に親切かつ丁寧な説明を施して、結果として負荷(A) を発生させている場合が多い。負荷(C) を減らすためには、使い方の学習をユーザの意思に任せて、かつユーザの期待通りの反応をする設計をしなくてはならない。

しかし、これは「ユーザを信じて、ユーザをコントロールしない」システムを設計せよ、という意味ではない。むしろ、ＵＩやマニュアルによって、「ユーザに気づかれないように、正しい使い方をするように誘導する」システムの設計を施すことで、負荷(C) を減らすことができるのである。

誘導されたことに気付かずに、ユーザ自身が自力で使い方を学習できた場合、ユーザはシステムを “Intuitive” （直感的）と評価し、使いやすさは [LEVEL 3] に達する。

以下、ひとつの機能に限って 負荷(A) (B) (C) を取り除く具体例を示す（図２）。

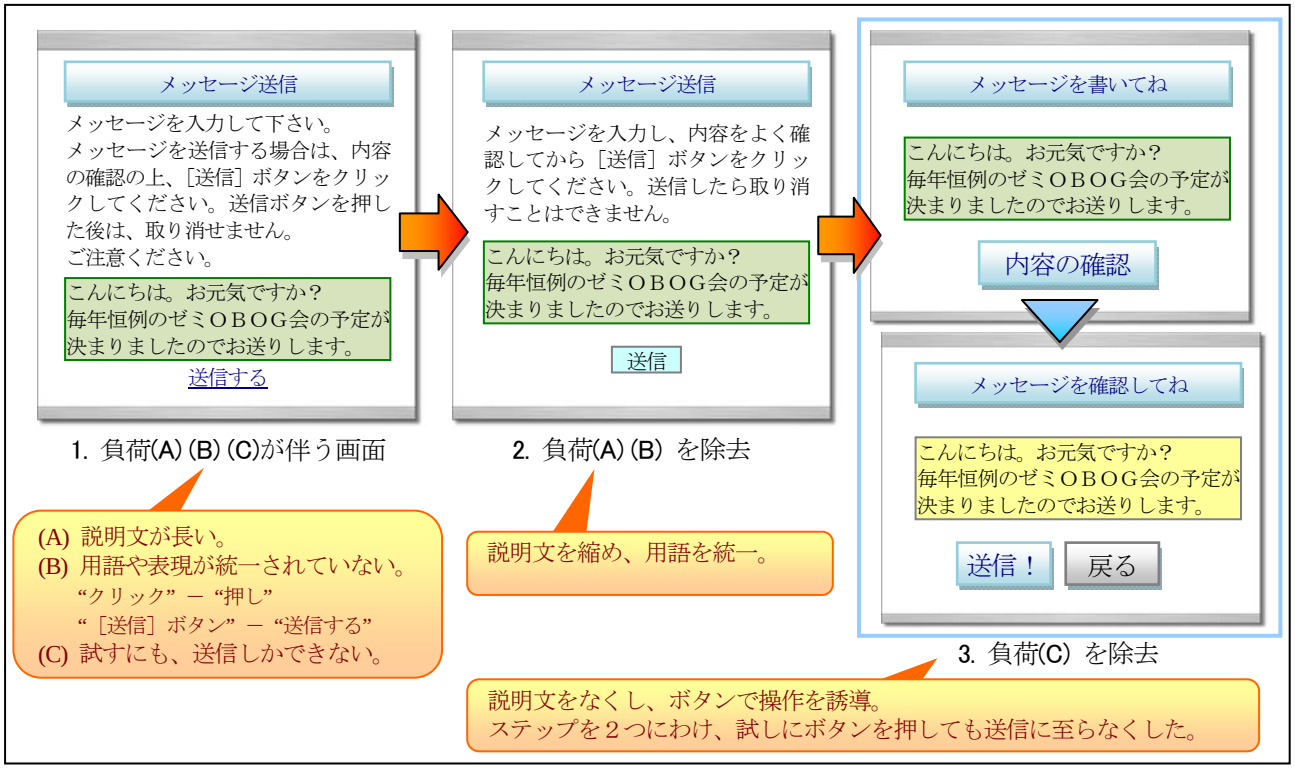

図 2　負荷(A)(B)(C)を取り除く具体例

6. おわりに

負荷(A) と負荷(B) を取り除く作業は、換言すれば「ユーザの興味を削がないように気遣い、ユーザが戸惑わないようにする」ということになる。これは、「人に対してやさしく」というアプローチである。一方、負荷(C) を取り除く作業は、「ユーザに考えさせて、試行錯誤もさせる」ということになる。これは「人に対して厳しい」というアプローチといえる。

前者のアプローチだけでは、「使いやすさ」は [LEVEL 2] までしか達成しない。 「使いやすさ」が [LEVEL 3] に達するには、「人にやさしい」だけでなく、「人に厳しい」アプローチも必要なのである。

参考文献

[1] 教育機関向け SNS 「manaba course」および 教育機関向けポートフォリオシステム「manaba folio」のホームページ。2007 年より提供。現在 34 校で導入されている。（2010 年 9 月時）
http://manaba.jp