

## 第5章 利用評価

---

第1節 ユーザー満足度

第2節 ユーザビリティ



## 第5章 利用評価

### 第1節 ユーザー満足度

---

ユーザー満足度はIT投資評価の主要な効果評価項目となっているが、その内容は深い意味と多様性を持っている。システム開発において

ユーザー満足度とは何を示すのか？

システム開発におけるユーザーとは誰を指すのか？

ソフトウェア開発の満足度モデルのコンセプトは、ハードウェアのそれとどこが異なるのか？

評価項目に何を取り上げるのか？

品質、工期、価額とユーザー満足度とはどのような関係にあるのか？

このユーザー満足度調査はどのように実施すれば良いのか？

等について順を追って考えてみよう。

- 1．顧客満足度とユーザー満足度モデルのコンセプト
- 2．基本サービスの評価項目と評価尺度の検討
- 3．表層サービスの評価項目と評価尺度の検討
- 4．プロジェクト責任者の満足度の検討
- 5．ユーザー満足度調査の計画と推進

## 概要・コンセプト

### = 検討経緯とユーザー満足度プロジェクトの意義 =

JUAS はどこか特定のベンダーに属さない立場で、ユーザー企業が集まり、企業の情報システム活動の活性化を研究・推進する協会である。ちなみに類似機能を持つ協会はフランスに CIGREF があるが、他には世界中にユーザーだけの協会・団体が存在している話を聞いたことはない。

では、ユーザーは何を満足させてもらえば良いのか？真正面から JUAS がこの問題に取り組み始めたのは 2001 年であった。

この頃はユーザー企業が情報子会社をそろって作り出した時期でもあり、「システム開発の問題は情報子会社の問題である」として親企業は関心を示さなかった。発注結果が高いのか、安いのか？品質が良いのか、悪いのか？何で測定するのか？は親の発注会社の重要な管理すべき問題であるにもかかわらず関心を持ってくれた企業は少なかった。

価額を含めた生産性、品質、工期は重要なシステム要素にもかかわらずなんら目安が無かったので「このような時期が長く続くはずがない」、「作りこみの発想で ISO、CMM に関心があるが、やがてその結果を評価する評価指標の重要性が認められるに違いない」等と先見性のある仲間を集めて「顧客満足度プロジェクト」を立ち上げた。

Capers Jones（「ソフトウェア品質のガイドライン」著者）が「ソフトウェア品質ガイド」を発行し品質基準について世界的に調査した結果を紹介したのは、2000 年である。彼は「日本は品質を重んじる国であり、他の国と比較しても際立って優れた国である」と認めてくれてはいたが、その中に紹介してあった品質実績の値は日本のユーザーでは認められないほどの低い値であった。

しかし、システムはより広範囲に活用され、その社会的影響は、はかりしれなくなっている。さて、ユーザー企業の方々と議論してみると、従来作成され利用されているさまざまな基準や考え方はベンダー主体の発想から出ているものが多く、ユーザーの目で見ても考え直したいことが数多く存在することも判明してきた。このユーザー満足度プロジェクトも顧客満足度プロジェクトとして発足した。まとめる段階に入ってから「ユーザー協会が顧客満足度を論ずるのはおかしい」との指摘がでてきた。顧客ではなく「利用者・ユーザー」であり、システム納品でなく「受け入れ」が正しいと言われてみればその通りで、直ちに「顧客満足度プロジェクト」は「ユーザー満足度プロジェクト」へと変身した。普段何気なく使っている言葉でも、立場が異なれば意味も違ってくる。

ISO9001:2000 によれば顧客重視の項目において「顧客満足の向上を目指して、トップマネジメントは、顧客要求が決定され、満たされていることを確実にすること」、また、監視および測定・顧客満足の項目においては「組織は品質マネジメントシステムの成果を含む実施状況の測定の一つとして、顧客要求事項を満足しているかどうかに関して顧客がどのように受け止めているかについての情報を監視すること。この情報の入手および使用の方法を決めること」と規定されているが、具体的な評価項目、標準値は規定されていない。

したがって、顧客は「この結果で満足すべきか？もっと良い方法・結果が世間では存在するのではないか？」などと疑問を持ちながらも我慢しているのが現状である。システム開発の世界では相当な数のシステムトラブルが発生して巷の議論の標的にされているが、今後もトラブル発生は避けきれないと思われる。

これらの背景を考察してみると、システム開発の世界においては「標準値、社会通念の常識値が存在しないことに起因するものが多い」。各種標準は「各開発工程でこのようなことをしなさい」と書いてはあるが目標値は与えていない。

後述するように JUAS のユーザー満足度は基本サービス、表層サービス、発注者満足度の3要素で押し量ることになっている。4年間経ての実情は、基本サービスは2004年度のソフトウェアメトリックス調査で工期・品質・生産性の実態が徐々にではあるが、判明してきた。しかしユーザー満足度と工期・品質・生産性の関係は十分に解明できたとは言えない。ユーザー満足度はまだ霧の中にあり今後の究明が必要である。

ユーザーに視点をおいたユーザー満足度のモデルを明らかにし、ベンダーがこの満足度モデルをユーザーの要求事項として認識し、ユーザーとベンダーの双方が共通の認識基盤を形成することにより、共通の標準値や社会通念としての常識値を持った、情報化社会の実現を期待している。

## 1. 顧客満足度とユーザー満足度モデルのコンセプト

システム開発についてのユーザー満足度調査方式を考える場合には、そのコンセプトが必要である。一般商品についての顧客満足度調査とシステム開発商品とは何が違うのか？一般商品の調査方式から学ぶことはないのか？と考え日本の顧客満足度を解説してある書物を数多く調査してみた。その中でこれなら活用できると見つけたのが以下の嶋口理論であった。

嶋口充輝（慶応義塾大学外学院 経営管理研究科教授）の理論  
（顧客満足型マーケティングの構図 有斐閣社 1994）

### （1）顧客へ提供するサービス属性の特徴

顧客がベンダーから提供をうけるサービスは大きく2つの属性に分けられる。

- ・基本サービス：顧客がベンダーへ支払う対価に対して当然受けられると期待しているサービス属性。
- ・表層サービス：対価にたいして、付加価値を顧客に期待させるサービス属性。

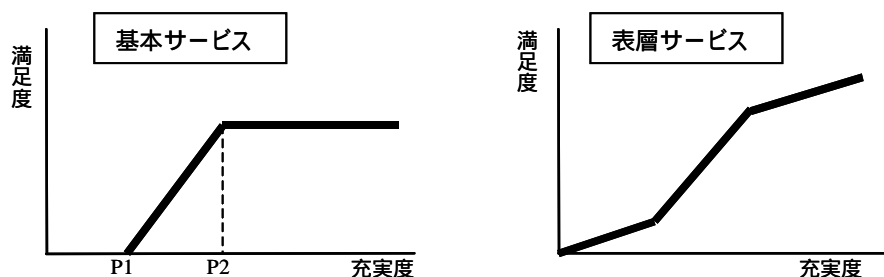
図表 5-1-1 は、企業が顧客に提供する2つのサービス属性に関する多様なサービス具体例であり、これら2つのサービス属性は顧客満足に異なる影響を与えられている。しかも、今日の充足された時代における顧客は、どちらか一方だけのサービス属性を充足させるのみでは十分な満足に到らず、両サービス属性がある水準以上に充足されないと全体満足が上がらない状況にある。たとえば、乗用車の場合、ただ基本性能や耐久性や安全性といった基本サービスがよいだけでは顧客は満足せず、それと同時に、デザインや色合いなどの表層サービス属性と相まってはじめて全体満足に結びつくという見方からも明らかである。

対象事業	基本サービス	表層サービス
銀行	<ul style="list-style-type: none"> <li>・ 安全性</li> <li>・ 確実性</li> <li>・ 公平性</li> </ul>	<ul style="list-style-type: none"> <li>・ 雰囲気</li> <li>・ 親切</li> <li>・ ユニフォームイメージ</li> <li>・ 美人</li> </ul>
乗用車	<ul style="list-style-type: none"> <li>・ 基本性能</li> <li>・ 安全性</li> <li>・ 耐久性</li> </ul>	<ul style="list-style-type: none"> <li>・ デザイン</li> <li>・ 色合い</li> <li>・ 各種アクセサリや装備</li> <li>・ ブランドイメージ</li> <li>・ 保証サービス 等</li> </ul>

（ユーザー満足度プロジェクト 2002）

図表 5-1-1 提供サービス属性の具体例

図表 5-1-2 は、企業による各サービス属性の充実努力と、それによって達成される顧客満足度の関係を図式化したものである。



図表 5-1-2 サービス充実度と満足度との関係

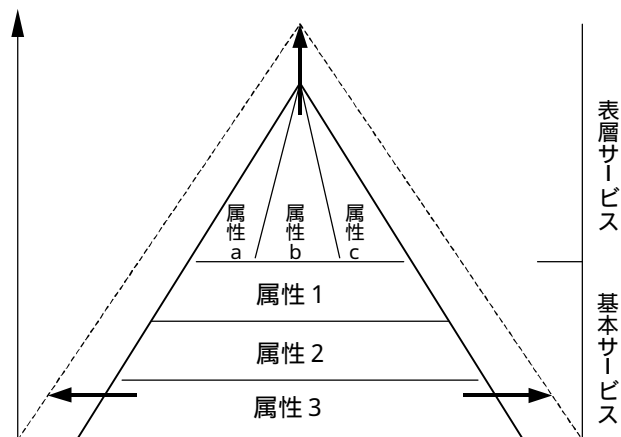
- ・基本サービス：

基本サービス属性の充実度がある最低許容水準（P1）以下のレベルになると、満足はゼロ以下（つまり、マイナスの満足 - dissatisfaction）になってしまう。充実度をあげていけば満足度は上昇するが、ある充実度レベル（P2）に到達すると満足度は水平状態になりそれ以上に上昇しなくなる。つまり、満足度がある一定レベルに到達するとそれ以上の評価は得にくくなることから、最低許容水準のサービス充実度（P1）が不満をつくらないという点で重要な意味をもち、この満足度水準を顧客とベンダーの双方で共通に認識し合意することが重要である。

- ・表層サービス：

付加価値を顧客に期待させるサービス属性であることから、このサービス属性がゼロであっても別に顧客は怒りの不満にはならず、単に満足していない（図中で満足がマイナスにならず、満足でも不満でもないゼロのまま - dissatisfaction）状態といえる。しかし、表層サービスの属性はそれらを充実させていくと、次第に満足水準が上昇していく特性をもっている。つまり、表層サービスは、顧客の満足度を直接向上させるための重要なサービス属性である。

(2) 基本サービスと表層サービスの構造



図表 5-1-3 満足のピラミッド (基本サービスと表層サービスの構造)

基本 / 表層サービスと顧客満足との関係を図 5-1-3 満足のピラミッドに概念的に表す。満足を大きくするには、底辺を広くして面積を大きくするか、あるいは、高さを上げて満足量を大きくするか、のどちらかである。

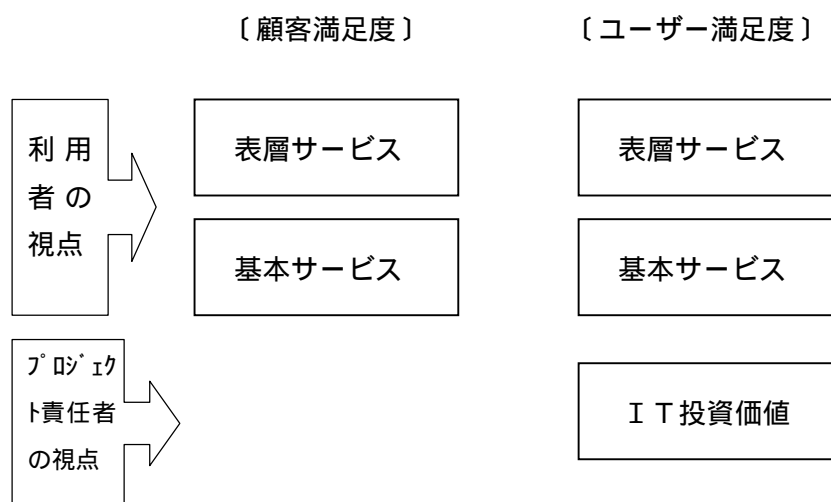
満足のピラミッドの構造は、その底辺部を基本サービスが下支えしている(ヨコ型の属性)。これは、基本サービスの属性のうち、どの一つでも最低許容水準を割ると、他のすべてが如何によくても全体満足が崩れ去り、不満になることを示している。いわば、属性間の代償作用がない重要なサービスである。

一方、表層サービスは、満足を上方に伸ばす役割を果たしており(タテ型の属性)、どれか一つでも卓越していれば、他の属性が悪くても、満足のピラミッドを高くすることが出来る。いわば、代償作業があるサービス属性である。

### (3) 顧客満足度とユーザー満足度

#### a. システム開発プロジェクトについての顧客満足度とユーザー満足度の違い

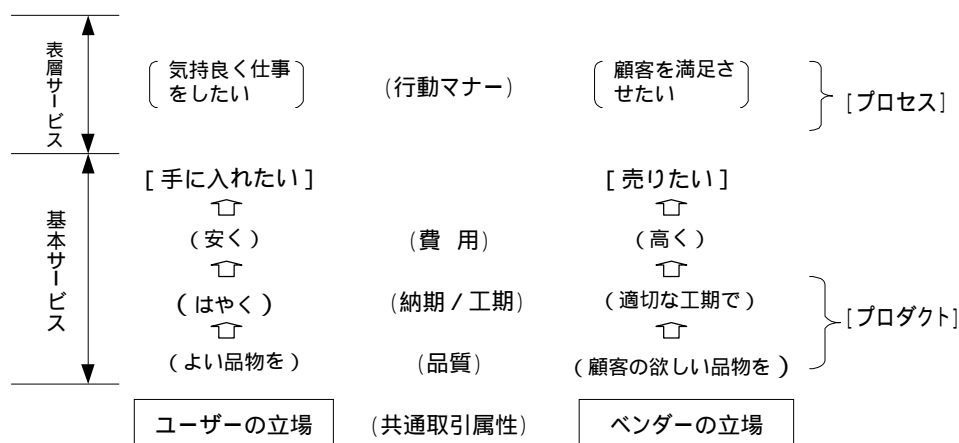
ベンダーは2つのサービス属性を通じて、顧客に製品を提供しその満足度に相当する対価をもって評価は完了するが、ユーザーは、2つのサービス属性だけではなく、さらにIT投資の価値も対象となる。これは、プロジェクト責任者によるIT投資が経営戦略の実現性/効果性にどの程度寄与したかの評価結果としての満足度である。つまり、利用者として(2つの)サービス属性の評価とプロジェクト責任者としての投資効果の評価とを加えた3つの評価要件が揃って、はじめてユーザーとしての満足度論が成立することになる(図表5-1-4)。



図表 5-1-4 顧客満足度とユーザー満足度

#### b. 基本/表層サービス属性の構成要素

満足度を考える場合、ユーザーとベンダーの双方が共通に認識しあえるサービス属性が必要であり、このサービス属性が共通の取引属性となる。ベンダー・ユーザー共通の認識下におけるサービス属性の構成要素を図表5-2-5に示す。



図表 5-1-5 サービス属性の構成要素

( a ) 基本サービス属性の構成要素

基本サービス属性としては取引の3要素である、品質、納期/工期及び費用が対象となる。具体的には、ベンダーが提供する価値としての「品質」と「納期/工期」、そしてユーザーが提供する対価としての「費用」から構成される。なお、基本サービスの納期/工期と品質は[プロダクト]を構成する要素でもある。

( b ) 表層サービス属性の構成要素

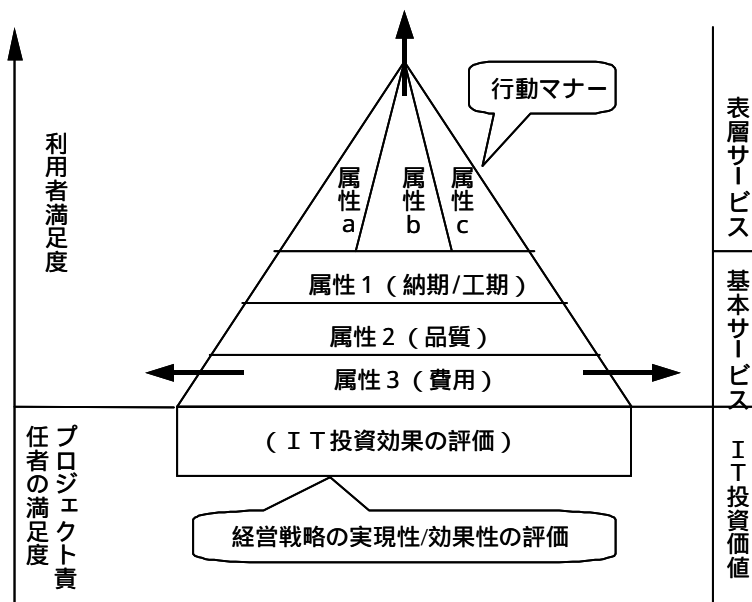
表層サービスの構成要素は「行動マナー」であり、ベンダーの管理者、営業、システムエンジニア(プロジェクトマネージャー)などが対象となる。

行動マナーは、自己認識(正確な自己評価、自己確信、他)、自己統制(信頼性、誠実性、他)、モチベーション(達成意欲、コミットメント、他)、共感性(他人への理解、サービス重視、他)、社会的スキル(コミュニケーション、リーダーシップ、他)など5つの領域に立脚する実践的スキルの学習能力を示唆するコンピタンスに依存している。コンピタンスはライフサイクルを通じて学習可能であり、経験から学ぶことを通じて向上しつづける特性を有している( \* )。

「行動マナーが本人の努力により限りなく向上する」ことは、「表層サービス属性の充実により満足度水準が上昇する(図表 5-1-2 参照)」という根拠をよくあらわしている。なお、行動マナーは[プロセス]を構成する要素でもある。

( \* : ビジネス EQ、ダニエル・ゴールマン、梅津祐良訳 東洋経済新報社、2000 )

c . ユーザー満足度コンセプトの構成要素と構造



図表 5-1-6 ユーザー満足度コンセプトの構成要素と構造

ユーザー満足度コンセプトの構成要素と構造を図表 5-1-6 に示す。基本サービスの 3 要素がピラミッドの底層部に、表層サービスの行動マナーが上層部にそれぞれ位置する。この行動マナーはプロダクトの全てのプロセスで、作りこみに作用しながら満足度を向上させていく。

IT 投資価値は基礎部（土台）に位置しており、プロジェクトの存立性をあらわしている。つまり、基本サービスと表層サービスの評価を受け止めて、IT ガバナンスによる経営戦略の実現性 / 効果性を評価していくことになる。

利用者満足度とプロジェクト責任者の満足度が有機的に組み合わせられて成立するユーザー満足論が当図のコンセプト構造の意味しているところである。

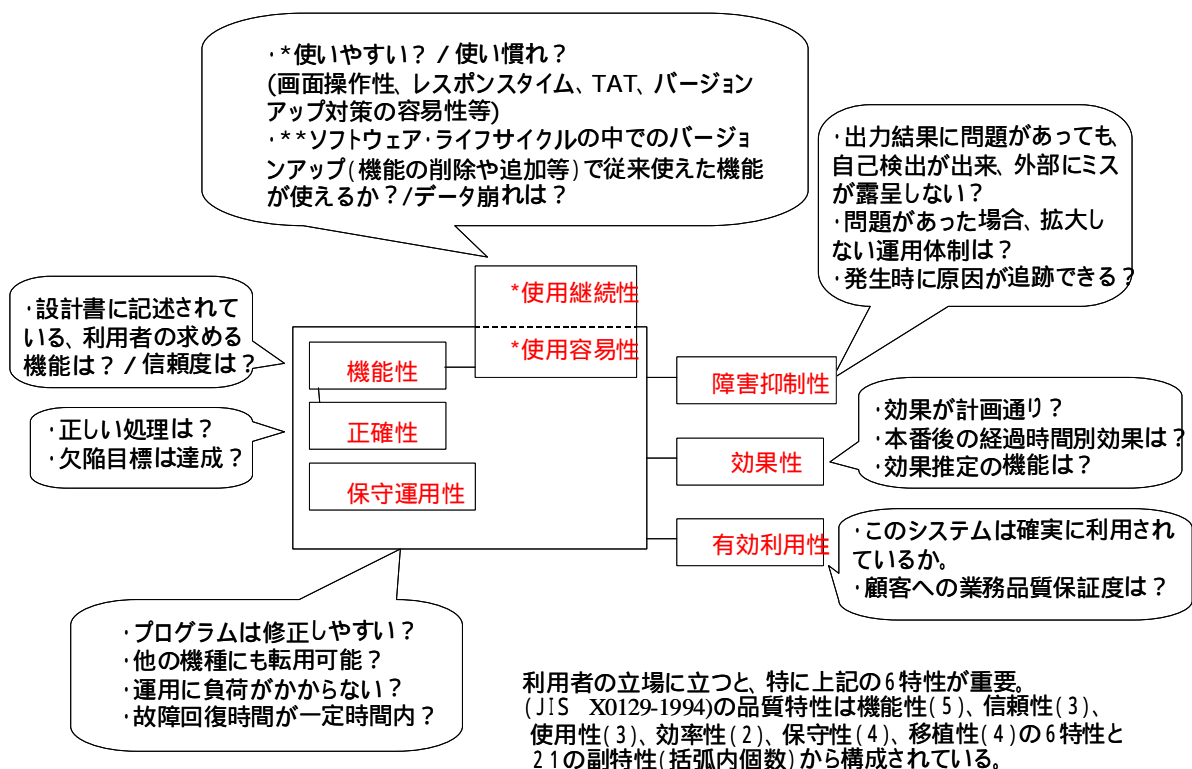
## 2 . 基本サービスの評価項目と評価尺度の検討

### ( 1 ) 品質の評価項目と評価尺度

#### a . 利用者オリエンテッドの品質特性

利用者の立場での品質は、ソフトウェアが網羅的に全ての必要性を特性化するのではなく、実務を通してソフトウェアの品質を実感・評価出来る特性値を対象とするものであり、それは6つの特性値（機能性、正確性、使用容易性・使用継続性、保守・運用性、障害抑制性、効果性）で構成される。

全体構造は、実業務の関連性から、結合性の強い 機能性 - 正確性 - 使用容易性・使用継続性のグループとそれとは独立に 保守・運用性の特性が枠内に在り、それらを囲う形でシステム全体を特性化する 障害抑制性と 効果性の2つが配置される（図表 5-1-7）。



図表 5-1-7 ユーザーの望む品質特性とは

ISO の品質特性との関係で言えば、枠内はそれに対応したものであり、枠外は今回ユーザーの視点で特性化した項目である。使用容易性の中に含まれる「使用継続性」については、システムは開発されたらそれで終わりではなく、企業の基幹システムの場合、日本では平均して 20 年保守を重ねて使い続けられる。同じようにベンダーのパッケージでも、バージョンアップを積み重ねながら、システムは成長する。機能が追加されて改善されるのは利用者にとっても良いことなのだが、場合によっては旧バージョンの機能が使えなくなったり、旧システムで登録されたデータをバージョンアップ後のシステムで利用しようとする、もともと正しく使えた機能が使えなくなったり、データの形が壊れたりし、同じ操作にもかかわらず、同じ結果が得られなくなることがある。このような不都合は避けて欲しいとする概念が使用継続性である。

「障害抑制性」は ISO の「障害許容性」にやや近く、「効果性」は強いていえば ISO の「合目

的性」に近いが、やはり異なる概念であるので枠外とした。特に「障害抑制性」は企業内システムにおいてサブシステム同士で結合し広い機能を持って活用されたり、企業間システムの結合で一つの大きな機能を提供したりする際でも、「たとえ他のシステムが間違っただータを提供しても、自社システムはそれに連動されない」等の自己防衛機能を一定限度内で備えるべき、とのユーザーの願望、期待を込めた特性である。

有効利用性は「システムは完成したが、誰も使ってくれないなどの事態に陥っていないかどうか?」「有効利用されている尺度には何を使うのか?」などのユーザーにとって重要な利用上の特性である。システムの出力のタイミング、内容の正しさなど利用者が迷惑を被っていないかどうか?を問う尺度も含まれる。

## b. システムの評価対象と評価項目

ソフトウェアライフサイクルプロセスでの評価対象と評価項目を図表 5-1-8 に示す。これらは、利用者の強いインセンティブというよりは、ベンダー自身が把握・評価しやすいように定めてきた傾向があり、ユーザーとして利用しづらいものが多い。

評価対象	評価項目
1. システムの欠陥性	
(1) バグ発生率	欠陥件数 / 開発費用、 欠陥件数 / LOC 数、 バグ件数 / FP 数
(2) バグ修正量	LOC 数、 FP 数
2. システムの性能性	
(1) サービスレベル管理	レスポンスタイム(オンラインリアルタイム処理)、 TAT(バッチ処理)
(2) ワークロード管理	スループット(トランザクション処理)、 資源使用時間 / 単位業務、 資源使用時間 / 月
(3) リソース管理	資源使用率、 アクセス回数、 資源の競合状態
3. システムの信頼性	
	信頼性(MTBF)、 可用性、 保守性(MTTR)、 保全性、 機密性

図表 5-1-8 システムの評価対象と評価項目

注・レスポンスタイム:端末でのデータ入力完了時から次の出力開始時までのトランザクション応答時間

- ・スループット:ジョブ数 / 時間、トランザクション数 / 時間
- ・TAT: Turn Around Time (ジョブ投入から完了までの時間)
- ・資源: CPU、DISK、回線、メモリ、チャネルなど
- ・資源使用率: 資源の使用状態時間 / 運用時間帯
- ・アクセス回数: 資源の使用頻度(ファイル出力回数、DISK I/O 回数、ページング I/O 回数など)
- ・資源の競合状態: 待ち時間、待ち合わせ回数、多重度
- ・MTBF: Mean Time Between Failures (平均故障間隔: 稼働時間 / 故障件数)
- ・MTTR: Mean Time To Repair (平均故障時間: 累積故障修復時間 / 故障件数)
- ・可用性: 稼働率 (MTBF / (MTBF+MTTR))
- ・保全性: データの保全を保証 (Integrity)
- ・機密性: データの機密性 (Security)

(データ出所: 21世紀へのソフトウェア品質保証技術 日科技連出版)

## c . 評価項目と評価尺度の検討

ベンダーよりの評価対象と評価項目から、ユーザーが品質状況を具体的に見ることができ、かつ、ベンダーと共通に認識できるものとして次の2項目を抽出した(表 5-1-9)。

No	評価項目	評価尺度	定義
1	システムの欠陥性 (正確性)	開発費用当たりの欠陥数  1件(欠陥数) (5百万円~1千万円)	<ul style="list-style-type: none"> <li>・欠陥数： 納品(受入れ)~安定稼働間で確認されたベンダー起因の欠陥数</li> <li>・金額： 基本設計~本番間のソフト開発費用</li> </ul>
2	システムの性能性 (使用容易性)	<ul style="list-style-type: none"> <li>・レスポンスタイム</li> <li>・TAT</li> </ul>	<ul style="list-style-type: none"> <li>・基準値(秒)</li> <li>・基準値(秒)</li> </ul> <ul style="list-style-type: none"> <li>・代表的入力作業が顧客の指定した基準値内に入る。</li> <li>・実用に耐え、かつ所定内に作業が完了する基準値。</li> </ul>

注 対象欄の( )内は利用者オリエンテッド品質特性

図表 5-1-9 品質に関する評価項目と評価尺度

システムの欠陥性に関する評価指標については、単位金額当りのものと単位規模(LOC、FP)当りのものがあるが、簡単でユーザーが判断し易くかつベンダーとの共通尺度として認識しやすい単位金額当りのものを評価項目とした(図表 5-1-9)。

但し、LOC、FP 単位の欠陥数で評価できる場合は、その使用を妨げるものではなく、併用も可能である。

項目	金額単位の欠陥数	LOC 単位の欠陥数	FP 単位の欠陥数
1.適用範囲と課題	<ul style="list-style-type: none"> <li>・大まか過ぎる点はあるが、開発言語、ツールに関係なく利用でき顧客にも分かりやすい。</li> <li>・パッケージを利用する場合にも相対的な比較の目安にはなる。</li> <li>・直営だけで開発する場合にも基準単価を設けて換算すれば利用可能。</li> </ul>	<p>一般に COBOL には適用されてきたが、言語、ツール、環境の多様化には適用しにくい。</p>	<ul style="list-style-type: none"> <li>・開発環境に依存することなく何れの開発言語にも対応可能。</li> <li>・ユーザー機能であることからユーザーにも分かりやすい。</li> <li>・パッケージ利用の場合は利用しにくい。</li> </ul>
2.品質の定義と目標	<ul style="list-style-type: none"> <li>・納入後、ユーザーによって発見された欠陥数を問う。</li> <li>・単体テストなどは一括発注するので、詳細データは顧客からは分からない。</li> </ul>	<p>テスト工数毎の目標値がベンダー毎に存在する。</p>	<ul style="list-style-type: none"> <li>・国際的な工数別実績あり。</li> <li>・ベンダー毎の目標値は存在する。</li> <li>・ユーザーとの共通目標値になり易い。</li> </ul>
3.プロジェクト毎の目標、組織毎の目標	<p>各プロジェクトは1件/5百万円以上発生させない。当初はユーザーとの契約値とはせず目標値でよい。</p>	<p>検収テスト（システムテスト）以降：0.1 個 / KLOC（例）</p>	<p>（0.01～0.03 個）/FP 以下の目標（例）</p>
4.コメント	<ul style="list-style-type: none"> <li>・スクラッチ開発のみならず、パッケージ+カスタマイズ及びプログラム保守作業にも利用できる。</li> <li>・ユーザーとベンダーの共通尺度として認識しやすい。</li> </ul>	<ul style="list-style-type: none"> <li>・レガシーの COBOL 中心時代には便利な尺度であった。</li> <li>・言語、ツール等多様化した環境には適用しづらい。但し COBOL との換算係数を使用して活用可能。</li> </ul>	<ul style="list-style-type: none"> <li>・クライアントサーバーの時代に適用し発達した。現在もスクラッチ開発システムには有効。</li> <li>・FP を利用しない顧客との共通目標にはなりづらい。</li> </ul>

図表 5-1-10 システム欠陥性の評価項目の比較

2004 年度に JUAS が実施したソフトウェアメトリックス調査の結果によると1件/500 万円を守れた品質のプロジェクトは約 40%であった。経験豊かなプロジェクトマネージャーが担当した場合は、ほぼこの目標がクリアできることも証明された。国際的には相当にレベルの高い目標値であるが、日本ではほぼ半数クリアできている。

もともと発注時に品質目標として、この値をベンダーに提示すれば、品質に伴うトラブルは大幅に避けられると思われる。ちなみに品質目標を提示していないプロジェクトの場合は、障害発生が多いことも証明されている。詳しくは「ソフトウェアメトリックス調査 2005」をご参照いただきたい。示唆に富んだ調査結果が数多く記載されている。

d . ソフトウェア品質管理のアクション

ソフトウェアは、ハードウェアと異なり、リコール処理をして部品を取り替えれば良いとする対策が一般的に使いにくい。従って、以下の二面作戦を推奨したい。

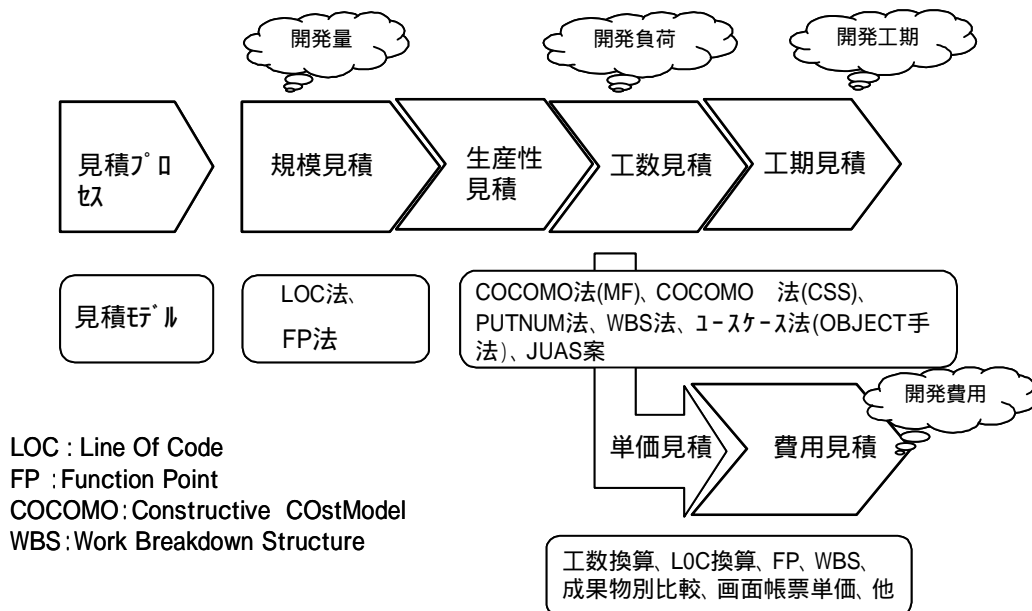
設定した品質目標値をベースに品質向上を期待する対策。

欠陥が発生した場合、ネットワーク等を利用して修復処理を早急に実施する。

(2) 見積プロセスと見積モデル

a . 見積プロセスと見積モデル

ソフトウェアライフサイクルにおける見積プロセスと見積モデルの基本構成を図表 5-1-11 に示す。規模見積プロセスの LOC 法、FP 法および工数・工期見積プロセスの PUTNUM 法、COCOMO 法、COCOMO 法は、何れも世界的に公知されたもので、実績プロジェクトのデータを統計的な処理によって計算式で表したモデルである。特に、FP 法はデファクト・スタンダードとしても利用されている。



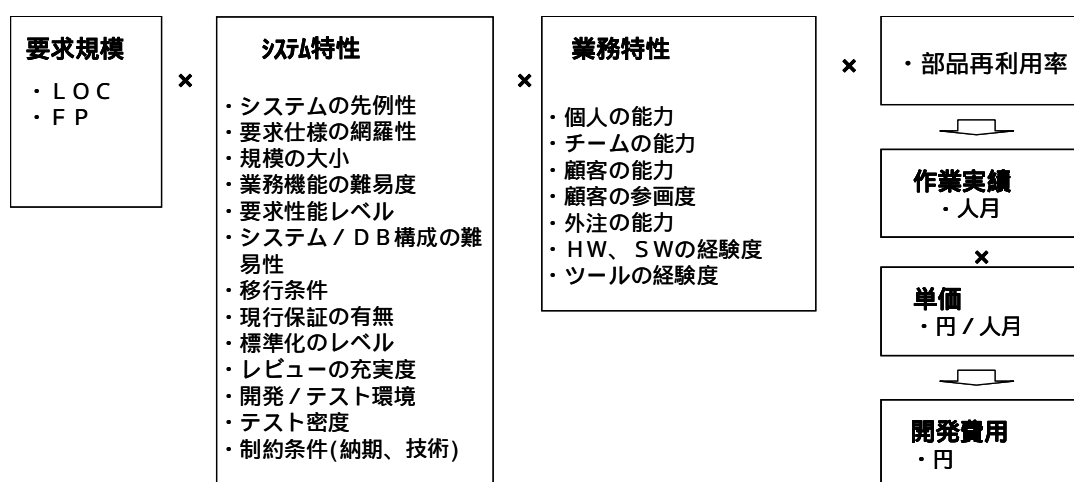
図表 5-1-11 見積プロセスと見積モデル

## b . 見積プロセスにおける変動要因への留意

見積プロセスにおいては、多くの変動要因が存在する（図表 5-1-12）ことから、これらシステム特性要因、業務特性要因を標準化し、定量的にモデルに反映させ、見積作業のバラツキ減少化に努める必要がある。

システム開発費用については、「この価格は妥当であろうか？」との悩みが常につきまとうものである。システム構造の簡単なものと複雑なものとは価格に差があることは理解できてもその妥当性を納得する尺度がない、あるいは適切な説明が不足しているので理解し難いものとなっている。

ベンダーとユーザーが共通の土台に立って話ができる見積方法、見積評価尺度の作成が必要である。上記コンセプトのシステム特性、業務特性を基準化しベンダー、ユーザーの共通評価尺度を JUAS にて作成したい。



図表 5-1-12 見積プロセスにおける変動要因

## c . 規模モデルの特徴

基本サービスの納期 / 工期と費用の 2 つの要素にかかわり、かつ見積プロセスの起点でもある規模モデルについて比較する。

### ( a ) LOC 法

- ・ 日本ではかなり以前から使用され、業界には普及・定着している。
- ・ ユーザーには、作業量（LOC 数）は判断しにくい。
- ・ 開発言語に大きく依存することから、代表的な開発言語である COBOL 等への換算が必要。
- ・ 当初機能に対し、SE がスキルをフルに発揮して LOC 数を少なく開発しても、ベンダーのメリットにはなりにくい。
- ・ 開発初期での適用は困難であり、過去の経験値を反映する必要がある。

### ( b ) FP 法

- ・ 開発言語に依存せずプロジェクト特有の機能を対象とすることから、ユーザーにとっては、開

発規模を客観的に理解・判断できる。従ってベンダ - と共通認識が可能である。

- ・要件定義時での概算見積、基本設計時の見積、詳細設計完了時の詳細見積を一貫して追跡できかつスケジュールへの適用を可能にさせるためには、現 FP 法の相当な改善が必要となる。
- ・内部処理に関する影響度の評価・適用の仕方については習熟が必要。
- ・パッケージ活用には適用しにくい。

### (3) 納期 / 工期の評価項目と評価尺度

システム開発規模（人月）から工期を算出する方法にはさまざまな方法があるが、工期短縮アクションの関係も含めて図表 5-1-15 に要約しておいた。

工期計算の細部には関心がない方は、(4)まで飛ばして読んでいただきたい。

#### a . ウォーターフォール型

##### (a) COCOMO モデルの課題

メインフレーム系・ウォーターフォール型のアプリケーション開発に適用される見積工数モデルは、1970 年代後半の PUTNUM(L.H.Putnum)モデルを経て、COCOMO(B.W.Boehm, 1981)モデルで一応の完成をみた。

COCOMO モデルは根本的な工数要因である開発規模とそれ以外の種々の工数変動要因とから工数及び工期を見積るための具体的な方法を与えるものであり、次の と を考慮した 9 つのモデル式で構成されている。プロジェクト初期段階のモデルは A・b の組み合わせである。

##### 対象とするシステムの見積り単位

- ・ A システム全体（プロジェクト単位）
- ・ B システム単位
- ・ C サブシステム単位 / モジュール単位

##### 対象とする業務

- ・ a 数日で開発できる簡単なフロント部のシステム
- ・ b 大部分のトランザクション処理システム
- ・ c 大規模で複雑なトランザクション処理システム / オペレーションシステム

通常は b が適用される。

##### プロジェクト初期段階のモデル

$$( ) \text{ 開発工数 (人月)} = 3.0 \times (\text{KSLOC})^{1.12}$$

$$( ) \text{ 工期 (月)} = 2.5 \times (\text{人月})^{0.35}$$

KSLOC : Kilo Source Line Of Code

##### プロジェクト初期モデルの課題

開発規模が LOC であることから、ユーザーは規模の妥当性を判断しづらく、FP やその他ユーザーが判断し易い規模モデルに置き換えるなどの手続きが必要である。

開発規模は、各企業・アプリケーション毎の性格によって、相当ばらつくので修正が必要となる。

##### (b) 評価項目の設定

プロジェクト全体工数（人月） = 自社の経験値（自社モデル）

通常、企業で使用されている言語類については、図表 5-1-13 に示すように、LOC から FP への変換が可能であることから、LOC、FP 何れの自社モデルも適用できる。

言語	COBOL	VB	C++	Java
SLOC数 / FP	100 (*)	70 (**)	50 (*)	35 (**)
換算係数	1.0	1.5	2.0	3.0

図表 5-1-13 FP と SLOC との換算

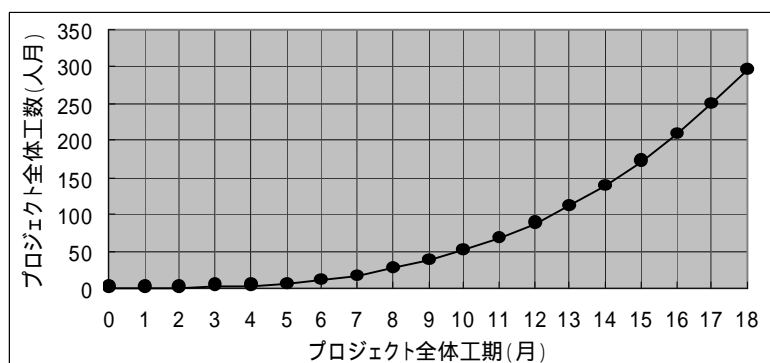
(\*) Software Quality Analysis and Guideline for Success by Capers Jones P191 共立出版社

(\*\*) 2002.10.17 開催の JUAS 講演資料

- ・ COBOL 換算 SLOC 数は、それぞれの言語における 1 FP 当りの所要工数比較比 × 100 で算出。
- ・ VB : 1.8(VB 工数対 1FP 工数)/2.6(COBOL 工数対 1FP 工数) × 100(COBOL・SLOC 数/FP)=69 70
- ・ Java : 0.9/2.6 × 100=35 35

$$\text{プロジェクト全体工期(月)} = 2.7 \times (\text{人月})^{1/3}$$

「ソフトウェアメトリクス調査 2005」の標準工期から計算したものである。これは COCOMO 法に則った形になっている。(図表 5-1-14)。



(ソフトウェアメトリクス調査 2005)

図表 5-1-14 JUAS の提案モデル

### (c) 評価尺度の検討

工期の設定においては、新商品の販売、対コンペ戦略、株式の上場、企業の統合などのタイミング重視プロジェクトにより必要工期が確保できないことが発生する。このような場合に対応できる工期推定方式が必要となる。

工期不足と対策の関係は、各企業で蓄積したものを活用するのが良いが参考として表 5-2-15 を掲げておいた。実際には金融業のシステム化ではこの標準工期よりも 20~30%長い工期が必要であり、流通業では逆に 20~30%短い工期の要請が多いことが経験的に判っている。なお、短工期の要請があったとしても、50%以下の工期での開発などは無謀である。

尺度	25%工期延長	標準	25%工期短縮	25%以上工期短縮
式	$3.4 \times (\text{人月})^{1/3}$	$2.7 \times (\text{人月})^{1/3}$	$2.0 \times (\text{人月})^{1/3}$	$1.4 \times (\text{人月})^{1/3}$
工期設定の考え方	金融等欠陥の発生を無くしたい品質重視	提案モデル	<ul style="list-style-type: none"> <li>顧客の要望</li> <li>流通業</li> </ul>	外的対応 / 工期制約 対コンペ戦略、新商品の販売、株式の上場、企業の統合など
工程管理上の対応策	十分なシステムテスト期間の確保	中日程計画の充実 (役割分担別WBS管理)	中日程計画の充実 (週間別管理)	小日程計画の充実 (日別管理)
その他の対応策	<ul style="list-style-type: none"> <li>品質重視のテスト計画書、テストケースの緻密化</li> <li>安定稼動のための分割立上等</li> </ul>	<ul style="list-style-type: none"> <li>WBSによる全体計画と局面化開発の徹底</li> <li>レビューの徹底</li> <li>テストケース充実</li> <li>コンバージョンデータのフル活用</li> <li>確実な変更管理</li> </ul>	同左 + (下記) <ul style="list-style-type: none"> <li>標準化の徹底と実力のある一括外注の採用。</li> <li>システム範囲、対象の部分稼動</li> <li>RAD+DOA</li> <li>性能事前検証</li> <li>変更管理の強化</li> </ul>	同左 + (下記) <ul style="list-style-type: none"> <li>ベテランプロマネの投入と会社あげての協力及び監視</li> <li>パート図での計画</li> <li>ベストメンバー選出</li> <li>クリーンルーム手法</li> <li>二交代制の配置</li> <li>開発促進ツールの採用</li> <li>顧客主体のテストチーム設置</li> <li>パッケージの活用</li> <li>部品の再利用</li> <li>オープンな進捗情報管理</li> </ul>

注 WBS : Work Breakdown Structure

RAD: Rapid Application Development

DOA: Data Oriented Approach

図表 5-1-15 工期評価尺度と対策

## b . スパイラル型

### ( a ) COCOMO モデルの課題

ソフトウェアプロセスが CSS<sup>1</sup>の場合、プロトタイプやスパイラル型が適用されているが、これは、メインフレーム系とは明らかに異なる下記のような特徴 / 課題があることに起因している。工数見積の変動要因ともなっているこれらの課題を解決しながら開発を進めざるを得ない CSS 特有のプロセス特性を列挙する。

#### 開発プロセス上の課題

新技術や未経験技術の導入、経験やノウハウの蓄積、開発要員の育成、ユーザーニーズに対応

<sup>1</sup> Client Server System: サービスの要求者 (クライアント) がネットワーク経由でサービスの提供者 (サーバー) の様々な資源を利用するオープン・分散型コンピューティング環境のシステム。経済性 (クライアントに廉価端末利用) システムの拡張性、サーバーによる共有業務 / 資源の一括管理、インタ・ネット利用による広域資源の活用等、様々な特徴をもっている。

した柔軟な変更などが要件である。

### プラットフォーム構築上の課題

プラットフォームはマルチベンダー提供によるプロダクトを組み合わせる。プロダクトの組み合わせによるプラットフォームはコンポーネントへの新技術適用や新コンポーネントの出現などにより標準化しづらく、しかもコンポーネントを構成するミドルウェアなどの寿命も短く、バージョンアップが頻繁に行われる。

### 開発期間の課題

システムの開発期間は一般に短い。このため、手戻りが発生するとリカバリーが難しく、重大なリスク要因になりかねない。

### 開発環境の課題

開発支援ツールをプラットフォーム上で動作させるために、両者のインターフェイスを確認する必要があり、ミドルソフトウェア、ネットワーク、データベースなどについて詳しい知識が必要になる。さらに GUI、4GL などの知識も必要。

### 仕様変更の課題

要求仕様に対しては、利用部門でも十分に理解しがたいことがあり、作りながらの仕様確定が行われることが多い。

COCOMO は COCOMO の改定版で、CSS に適用されるスパイラル型や成長型などの新しいソフトウェア開発のパラダイムに対応するために開発された見積モデルである（南カルフォルニア大学アーバイン校及び 28 企業から構成されるプロジェクトチーム、1997 年）。

これは図表 5-1-16 のモデル式で構成され、プロジェクト初期においては、の初期設計モデルが対象となる。

モデル	概要	適用
アプリケーション組立モデル	・規模に基づく線形モデル	プロトタイプ工数の見積
初期設計モデル	・ファンクションポイントの採用 ・規模、規模変動要因（5 個）、工数変動要因（7 個）に基づく非線形モデル	設計の初期段階での見積
ポストアーキテクチャモデル	・ファンクションポイント / SLOC の採用 ・規模、規模変動要因（5 個）、工数変動要因（17 個）に基づく非線形モデル	詳細設計以降での見積

図表 5-1-16 COCOMO モデルの概要

$$( ) \text{ 開発工数 (人月)} = 2.5 \times (\text{KSLOC})^{1.01 + W_i} \times E_i$$

$$( ) \text{ 工期 (月)} = 2.7 \times (\text{人月})^{0.33 + 0.2 \times W_i}$$

- ・ KSLOC : Kilo Source Line Of Code
- ・  $W_i$  : 規模変動要因
- ・  $E_i$  : 工数変動要因

規模変動要因及び工数変動要因はそれぞれ図表 5-1-17、図表 5-1-18 で構成されている。

	レベル (最大 5% の高低差)
--	------------------

規模変動要因 (W <sub>i</sub> )		レベル					
		非常に低い	低い	平均的	高い	非常に高い	極めて高い
W1	開発の先例性	0.04	0.03	0.02	0.02	0.01	0.00
W2	開発の柔軟性	0.06	0.05	0.03	0.02	0.01	0.00
W3	アーキテクチャ/リスクの理解度	0.04	0.03	0.03	0.02	0.01	0.00
W4	チームの凝集度	0.05	0.04	0.03	0.02	0.01	0.00
W5	プロセスの成熟度 (CMM)	0.05	0.04	0.03	0.02	0.01	0.00

図表 5-1-17 規模変動要因

工数変動要因 (E <sub>i</sub> )		レベル (最大 151%の高低差)					
		非常に低い	低い	平均的	高い	非常に高い	極めて高い
E 1	開発要員の能力	1.12	1.08	1.00	0.94	0.87	0.75
E 2	製品の信頼性と複雑性	0.89	0.94	1.00	1.05	1.17	1.34
E 3	再利用性要求	-	0.91	1.00	1.14	1.29	1.49
E 4	プラットフォームの困難性	-	0.96	1.00	1.06	1.22	1.51
E 5	開発要員の経験	1.15	1.07	1.00	0.94	0.89	0.82
E 6	開発ツール使用性/マルチサイト開発	1.18	1.11	1.00	0.89	0.79	0.75
E 7	開発スケジュール要求	1.29	1.10	1.00	1.00	1.00	-

図表 5-1-18 工数変動要因

### (b) スパイラル型適用プロジェクトの調査

COCOMO の初期設計モデルは、規模の変動要因及び工数の変動要因ともかなり複雑な変数を反映した構造となっており、プロジェクト初期の全体感を把握する段階では、変動要因の内容すら判明していない状況であることから、初期設計モデルを適用することはできず、少なくともウォーターフォール型に見られる JUAS モデル程度の簡単な計算式で算出することが望まれる。

スパイラル型のモデルを追及するために、下記の要領での実態調査が考えられる。

評価のポイント：スパイラル型の効果（対ウォーターフォール型）

調査対象工程：基本設計～本番

調査項目

- ア．実工数をベースとして実際の工期、生産性及び品質を比較。
- イ．実工数をベースとした実工期比を抽出。
- ウ．JUAS モデルの適用可能性。
- エ．代替式の必要性和必要時の新モデル算出の可能性。
- オ．契約との関係（繰り返し結果の見積りへの反映）

## (4) 評価項目間の相互関係の検討

### a. 品質と費用間の相関関係

#### (a) 品質と費用間の課題

業務用に開発したプログラムにバグが多く含まれておりその対応に追われたユーザー、あるいは欠陥のあるパッケージを購入し困った経験のあるユーザーは多い。

一方、「使用に耐えられるソフトウェアであれば使いながら修正する」ことも現実に行われており、「必要以上に高度な品質を要求することは、高い価格のソフトウェアを買うことになる」のも事実である。ではどの程度高度な品質を要求すればどの程度費用が増加し結局利用者は高価なソフトウェアを購入することになるのか？と尋ねても明確な回答が出きるベンダーは、この情報化社会でも、ほとんどない。

ユーザー企業の経営者は増加し続ける IT コストの削減に躍起である。少しでも安いソフトウェアが入手でき、かつ安心して利用可能であれば満足してくれることは間違い無い。

ハードウェアについては JIS その他の規格があり日本の商品の品質は世界でも最優秀との評価を得ている。ソフトウェアについては目標が無いことが問題である。

ベンダーは、発注者の承認を得た仕様に基づきプログラムを作成するのが通常である。ユーザーはテスト結果を確認してから利用開始したはずである。それでも問題が発生すると品質への要求は厳しさを増してくる。

仕様の提示、承認、テストデータの提供、テスト結果の確認など利用者自らが分担しなければならない業務は多い。発注した結果のソフトウェアが納入された場合の品質は利用者・発注者の相互努力の成果である側面を持っている。

しかし残念ながらソフトウェアの品質のレベルと価格（費用）の関係はベンダーが提供するのを嫌がること、品質に及ぼす影響要因の多様性と分析不十分などの理由により明確でない。ユーザーは「利用者もなすべき努力は果たすが、ベンダーも品質と費用の関係を明らかにして欲しい」と要求し、相互の努力により高品質低価格のソフトウェアを利用者が入手できることを JUAS としても推進したいと考える。

#### (b) 品質と費用の関係

品質と費用の相関関係には、品質尺度を X 軸に、欠陥発生状況と費用を Y 軸に、そしてユーザー満足度をパラメーターとした場合、図表 5-1-19 のような関連性が想定される。以下にコメントを列挙する。

目標品質の上昇と費用の関係については、ほとんどのベンダーが明確な基準（金額当たりの欠陥個数）をもっていないようである。

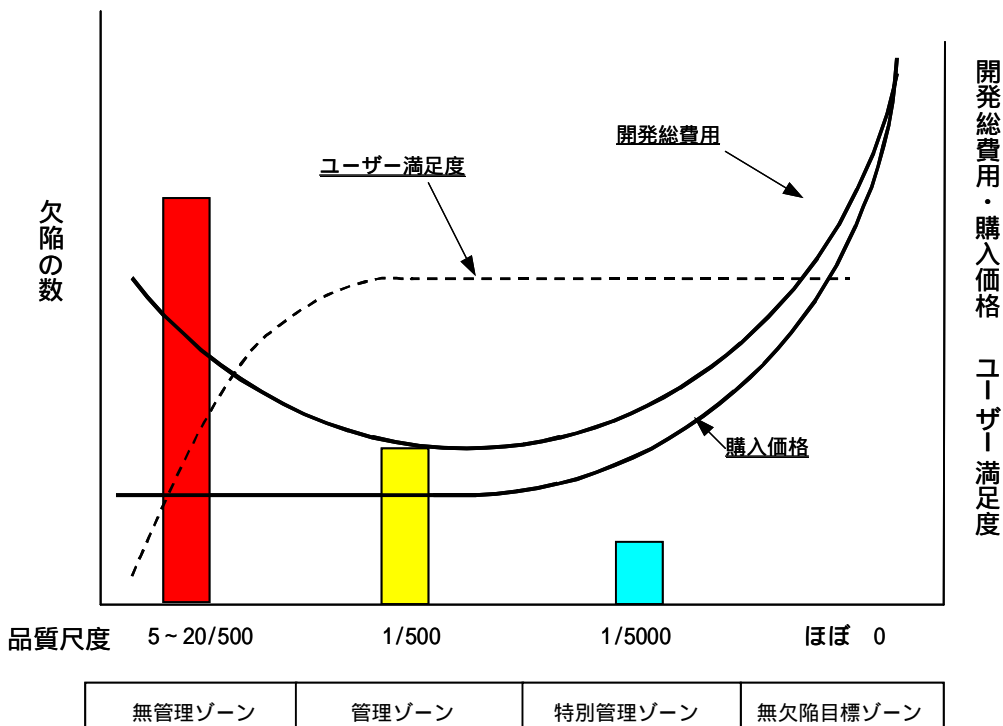
厳しい高度な品質をユーザーが要求すれば、当然ベンダーは、特別管理ゾーン、無欠陥目標ゾーンに品質目標が入るように厳しい開発プロセスを採用せねばならず購入価格はアップする。ライフライン用のシステムなどはこの無欠陥目標ゾーンを採用せねばならないが、ユーザーとベンダー相互の努力により実用可能なレベルに落とし込める場合は必要以上の精度を要求しないほうが、安く必要機能を入手でき、開発総費用も低下する。

一方ユーザーとベンダーの双方に品質の目標が設定されていない場合は「無管理ゾーン」に陥

る可能性があり、ユーザーは欠陥発見のための検査、修正後の再検査に負荷を取られることになる。結局高い買い物をしたことになる。ユーザー満足度も低い。

ユーザーにとっての欠陥に関する品質は「プログラムを開発者から受取り、納品検査を実施し、システムテストをした後、本番を迎え安定稼働迄に発生した欠陥の数」で評価する。本番稼働後に発生した欠陥数だけで評価するのではない。つまり、「プログラムを納入する時には、一定率以上の欠陥をつけてもって来ては困ることを宣言している」ことにベンダーは着目されたい。ベンダーの各社に「仮に貴社で 500 万円に 1 件の品質保証ができている場合に、5,000 万円に 1 件の障害に抑えて納入してくださいと顧客から要望が来たら、何倍の費用をもらえば対応が可能ですか？」と質問してみる。20%増、50%増、100%増、200%増、500%増と区分して挙手をして反応を見ると、ソフトウェア開発各社のレベルが良く分かる。手を上げられない企業は品質について日頃のしつけができていない。2 倍、5 倍で手を上げる企業も日頃の品質が概して悪い。既に相当なレベルの品質管理を実施している企業は 20%アップで手を上げる。このようなベンダーのレベルを発注側の企業はどのように判断しているのだろうか？

品質尺度と費用の関係を広く調査して、品質各ゾーンの境界の臨界精度とアクションの関係を見極めていかなければならない。



注1 品質尺度：(納入時～安定稼働期迄の欠陥個数) / 開発費用(万円)

注2 開発総費用と購入価格のギャップはテスト結果の確認、修正結果の確認のために要するユーザー側の負荷増加費用をイメージ化したもの。

図表 5-1-19 品質と費用との関連性

## (5) 工期と品質間の相関関係

### a. ユーザーの品質への要望と開発生産性への影響

ユーザーによる品質要望はベンダーの作業負荷見積や開発生産性へ大きく影響を及ぼし、結果として工期・工数を変動させる要因となる。工期に対する品質の特性要因を十分に点検・吟味して、作業負荷見積や開発生産性に反映させ、両者の相関関係を定量的に明確化する必要がある。

品質要望に対応する作業負荷見積と開発生産性標準の実例を図表 5-1-20 及び図表 5-1-21 に示す。

品質特性	尺度項目	数量化方法	影響水準
機能性・合目的性	要求仕様の網羅性	要求仕様書の記述水準を評価する。	2 : 95 以上 ~ 100% 3 : 75 ~ 95% 未満 4 : 50 ~ 75% 5 : 50% 未満
機能性・正確性	レビュー充実度	レビューに使用する時間と全作成工数の比であらわす。	2 : 1.0 倍 (10%) 3 : 1.5 倍 4 : 2.0 倍 5 : 3.0 杯以上
	テスト密度	テスト項目の設定密度	2 : 1.0 倍 (10%) 3 : 2.0 倍 4 : 4.0 倍 5 : 8.0 倍以上
	付加作業数	作業リスト例 ・連続作業立会 ・全項目再試験 ・性能試験 ・付加試験	

図表 5-1-20 品質への影響と作業負荷見積 (株ジャステック資料)

工程別生産性 ドキュメント	基本設計 H/P	詳細設計 H/P	コード化 (*)	単体テスト H/項目	統合テスト H/項目	システムテスト H/項目
基本ドキュメント	7.0	5.0	40.0	-	-	-
・テスト仕様書 ・テストデータ 作成 ・テスト実施 ・検証, 報告	システム	10.0	10.0	30.0	-	2.0
	統合	12.0	6.0	30.0	-	3.0
	単体	10.0	3.0	30.0	1.0	-

(\*) 基本ドキュメント : H/KLOC、 テスト : H/KC

図表 5-1-21 開発生産性標準 (株ジャステック資料)

### 自社実績データの具体的反映のステップ

品質要望に対応する作業負荷見積 (表) と開発生産性標準 (表) の実例をベースとして、自社実績データの具体的な反映のステップを示す。

( ) 適切な尺度項目を 50 項目程度準備し、ベンダー・ユーザー双方の努力レベルや対策な

どについて十分に確認し認識を合わせる。

- ( ) 尺度項目の数値化が可能のように、影響水準に統一的評点{ 評価 2( 1.0 ) 評価 3( 1.05 ) 評価 4 ( 1.15 ) 評価 5 ( 1.20 ) } を付ける。
- ( ) 尺度項目別、品質特性別に評価点を加算して品質を数量化する。
- ( ) 成果物納入時に、1 件 / 500 万円程度の欠陥しか発生させないとして、図表 5-1-21 の尺度項目と比較し表の妥当性を評価する。
- ( ) 評価結果を基に図表 5-1-20 の適切性を検討する。

図表 5-1-20 の開発生産性標準での品質よりも更に良い品質を要求される場合には、仕様書の作成密度 レビューの充実度 テスト密度 負荷作業数 ( 全項目再試験、性能負荷試験、立会テストの程度 ) 外部検査機関数 ( ISO 使用有無、他 ) テスト協力度 ( ユーザーテストデータ作成、検証、などに協力する度合いが高ければ開発者負荷は減少 ) 指定されたソフトウェア ( パッケージ、ミドルソフトウェア、ケースなどの安定度 ) 計算機使用可能度などの条件について影響度を把握し特性値分析を重ねる。

上記記述内容は「(株)ジャステック」の資料を参考とした。

## b . 工期と品質の関係

工期と品質との相関関係には、工期尺度を X 軸に、欠陥発生状況と生産性を Y 軸に、そしてユーザー満足度をパラメーターとした場合、図表 5-1-22 のような関連性が想定される。

以下にコメントを列挙する。

工期の変動と費用の関係についても明確な基準が存在していないようである。

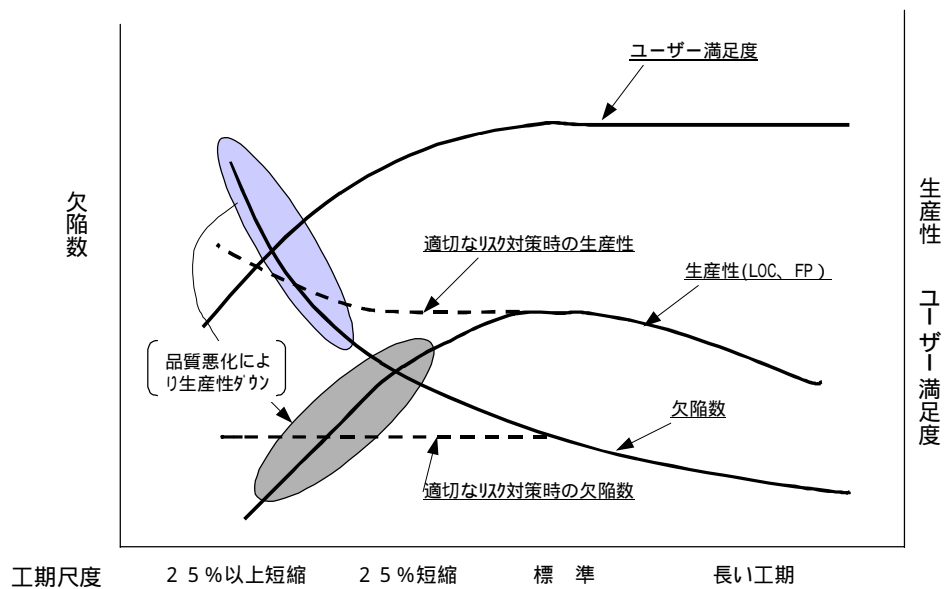
金融システムなど欠陥の発生をなくしたい品質重視の場合には、標準よりも長い工期を確保し当初の品質を作りこむことになる。

年末商戦、対コンペ戦略など外的要因により工期を大幅に短縮せざるを得ない場合があり、当然投入要員も増強することになる。この場合、単に頭数だけを増やしてむやみに投入しても要員間の連携、個人の能力差による進捗遅れなどが起こり生産性ダウンや品質悪化が発生することが多い。ユーザー満足度も低い。

一方、事前に要員増強の必要性を十分に把握し、ベテランプロマネによる采配と会社挙げての協力、スキルフル要員のタイミングよい投入、各種開発ツールの適用、適切な管理ツールの適用・オープンな進捗管理などにより品質を低下させることなく生産性を向上することも可能である。ユーザー満足度も高い。

かくなる状況とそのリスク回避策をユーザーとベンダー間で共通認識する必要があり、そのためにもベンダーから積極的な提言が必要で、それに呼応してユーザーも協力することが重要である。

工期尺度と生産性、品質の関係を広く調査し、工期尺度と汎用的なアクションの関係を見極めていかなければならない。



図表 5-1-22 工期と品質との関連性

### 3 . 表層サービスの評価項目と評価尺度の検討

#### (1) 表層サービスの調査・分析

表層サービスは、ソフトウェアライフサイクル上、プロセスとして位置づけられ、基本サービスのプロダクト（品質、納期）を的確に作りこむうえで重要なサービスである。

表層サービスの評価項目を設定するために、顧客満足度調査アンケートの質問項目を、数社調査した（図表 5-1-23）。調査に際しては、質問の意図 / 内容を明確に把握するために項目数 63 の対象を三者に層別した。

a : 主として顧客のための質問

b : 主としてベンダーのための質問

c : 両者のための質問

	管 理 者	営 業	SE (PM を含む)
営業活動	a : 顧客の経営的課題を視野に入れた提案をしていますか？ a : 顧客への関心は十分でしたか？ a : 適切なタイミングで顧客訪問を行いましたか。 a : ステークホルダーの意向を十分に把握しましたか？	b : 高い訪問頻度がなされていますか？ b : 常時連絡のとれる体制ができていますか？ c : 業種 / 業務に関する知識は十分ですか？ c : 製品技術に関する知識は十分ですか？ a : 情報の提供 / 助言は十分になされましたか？ a : 問題解決のための提案をしましたか？	b : 常時連絡のとれる体制になっていますか？ b : ご要望の理解、的確な対応ができましたか？ b : 業種 / 業務に関する知識は十分ですか？ b : ハード / ソフト製品に関する知識は十分ですか？ b : 先進製品技術に関する知識は十分ですか？ a : 情報の提供 / 助言は十分になされましたか？ a : 適用業務の分析 / 企画 / 提案は十分でしたか？
RFP回答	a : 内容は品質の良いものですか？ a : 内容はわかりやすいものですか？	c : プレゼンテーション技術は高いですか？ c : タイムリーの良い見積作成がなされましたか？	c : 説明は親切でしたか？ a : 内容は十分で納得のゆくものでしたか？
契約迄	c : 実行範囲が明確になっていますか？	b : 注文 / 請求 / 契約の手続きは判りやすいものでしたか？ b : 注文 / 請求 / 契約の手続は正確なものでしたか？ b : 注文 / 請求 / 契約手続の実行の際は適切な対応をとっていましたか？ b : 要求事項に対する把握 / 分析 / 提案は十分でしたか？	
設計時	b : 顧客の要望を正しく理解し部下を指導していますか？ a : 問題発生時には直ぐにかけつけますか？	b : SE との連携は緊密でしたか？ c : 問題点の把握は確実でかつフォローを十分にしましたか？	b : 経験豊かで業務内容を熟知していますか？ a : パッケージを熟知し最少の改造 / 追加の提案ができましたか？ b : 先端技術に詳しくかったですか？ c : 移行後の問題解消のための検証まで配慮した設計をしましたか？

	管 理 者	営 業	SE ( PM を含む )
開発時	c : タイミング良い訪問がなされ問題を共有しましたか？	c : 仕様変更に対するマナーは納得のいくものでしたか？ c : リスクの早期発見に努め、問題発生を未然に防ぎましたか？	c : PM の技術力とリーダーシップは十分でしたか？ c : 約束は厳守しましたか？ b : 幅広い業界知識を持っていましたか？ c : 経験の豊かさを感じさせるコミュニケーション能力がありましたか？ a : 問題発生時の対応が的確 / 迅速でしたか？ c : 計画通りに進捗させる力は十分でしたか？ c : 適切な報告がされましたか？ c : 適切な仕様変更をとりましたか？ c : パッケージの活用技術力は十分でしたか？
本番	c : 実態の正確な把握と問題発生時の迅速 / 的確な対応は十分でしたか？	c : トラブルシュートに対する対応は的確でしたか？	c : トラブルシュートは迅速かつ正確でしたか？
保守	c : SLA を持ち顧客の満足を得る努力を十分にしましたか？ c : サービスのために受信人払いの電話を十分に用意してありますか？ c : サポートセンターへの苦情は適切に扱われていますか？	c : パッケージのバージョンアップの頻度は適切でしたか？ a : 機能の継続性を保証していますか？ a : バグや欠陥の報告と対応は十分にされていましたか？	c : 修正に対する見積りは妥当なものでしたか？ a : 修正は迅速かつ正確になされましたか？ c : 納期を遵守する姿勢と努力は十分でしたか？ c : 適切な作業の進め方と報告がなされましたか？ c : 保守作業プロセスに改善の余地はありませんでしたか？ a : 緊急時の体制構築は十分でしたか？

注 管理者とは営業、PM の上位に位置するものをさす。ハードウェアの製品を導入、調整するハードウェア周辺サポートの SE については今回対象から外した。(合計：63 件)

図表 5-1-23 顧客満足度調査アンケートの質問項目 (開発・保守業務対象)

#### コメント

技術力とコミュニケーション能力の組み合わせが多い。

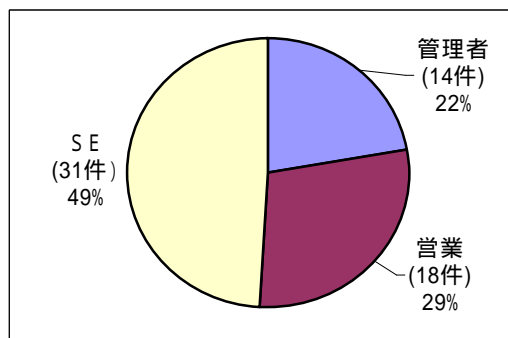
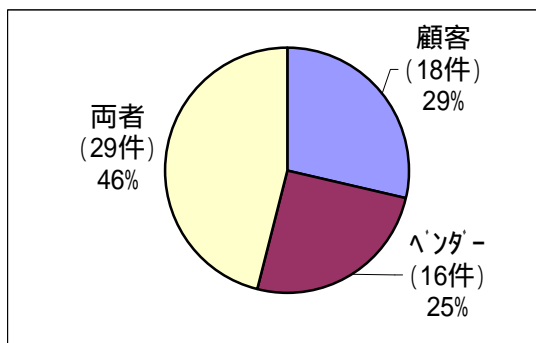
質問結果が改善アクションに結びつきやすいものと調査、検討後でないアクションが見えにくいもののが混在している。

質問項目はフェーズ別に区分されており、フェーズに特化した業務に基づく行動性がよく認識できる。

質問項目を対象別に分析したところ、顧客とベンダーに対してはほぼ同じレベル(顧客(a): 25%、ベンダー(b): 29%)であるが、両者(a + b)にかかわる質問は 46%とかなり多い。これは、仕様書型式が機能仕様書であり、仕様に対する双方の認識合わせと合意によって(両

者がかかわり合いながら)仕事を進展させるというプロジェクト特性に起因している(図表 5-1-24)。

質問項目をベンダーの関係者別に分析したところ、管理者に対する質問は営業と同レベル(管理者:22%、営業:29%)である。営業と同じように管理者の責任も重いことが窺える。なお、SE( PM 含む)に対しては、49%と大きくプロジェクトの成否の根幹をなす位置づけにあることが確認できる(図表 5-1-25)。



図表 5-1-24 対象別質問項目数比率

図表 5-1-25 ベンダー関係者への質問項目数比率

上記表は顧客満足度の実施に際して表層サービスについてのチェックリストとして利用できる。

## (2) 評価分類の検討

プロジェクト完了時に、ユーザー/ベンダー双方が、評価をどのように分類するかについて、次に示す。

### a. 「ユーザーの評価内容」に応じた、ユーザー/ベンダーお互いの感じ方

評点	ユーザー評価内容	ユーザーの思い	ベンダーの思い
1	今後お付き合いしたくない	プロジェクトは失敗であった。顔も見たくない	ユーザー側の責任もある。今後お付き合いしたくない顧客である
2	多くの候補会社の一つとして考える	特別に良くも悪くもない。新規プロジェクトは白紙で考えた	平均点の評価であり、通常のお付き合いが願えると思う
3	担当者が変われば付き合いってもよい	開発会社は悪い会社ではないが、今回は担当者が良くなかった。担当者、責任者を変えてお付き合い願いたい	今回は評価して貰えなかったが、今までの良い評価の実績がある。担当者、責任者を変えるのでお付き合い願いたい
4	リピート注文をだしても良い	プロジェクトは成功したので、この次も発注したいと考えている	是非次回の受注もお願いしたい
5	ひいき顧客になってもよい	<ul style="list-style-type: none"> <li>今回のプロジェクトを含め、どのプロジェクトも成功し満足度が高いので、他の分野の仕事もこの会社に依頼したい。</li> <li>当社の IT 部門支援の一翼を担ってほしい</li> </ul>	<ul style="list-style-type: none"> <li>素晴らしい顧客であり、ひいき顧客として確保したい。</li> <li>他社への紹介が期待できる</li> </ul>

図表 5-1-26 プロジェクトの評価分類表

## b . お互いの期待と評価

リピート注文を頂くのは、満足度の第一段階であり、一歩進めてひいき顧客にまでなって頂くのが最高である。

当然のことながら、発注者の評価値と受注者の評価値が同じ点になるとは限らない。

新規プロジェクトを始める場合、「新鮮でかつ有効な提案がある」ことが第一条件であり、その次に「確実な実行力があり、ユーザー満足度を確実に確保できる」ベンダーが、発注者側から期待されている。

お客様の6段階：お客様の分類を区分して見ると次の6分類になっている。

suspect(潜在顧客)、prospect(見込顧客)はマーケティング段階の顧客概念であり、buyer(お客)、customer(顧客)、client(馴染顧客)、advocate(ひいき顧客)が販売段階の顧客である。

suspect(潜在顧客)、prospect(見込顧客)、buyer(お客)、customer(顧客)、client(なじみ顧客)、advocate(ひいき顧客)

(参考：「貴方が創る顧客満足度」 日経ビジネス文庫 佐藤知恭著)

新しい顧客のシステムを開発するのに際して、上司はどの段階のお客になって欲しいのか？SEに伝える必要がある。

## c . ベンダー管理者への期待

ベンダーの若いSEに「新しい顧客のプロジェクトを開始する場合に、貴方の上司は何と言って貴方を送り出しますか？」と質問をさまざまな場でしている。

何も特別な指示はもらわない。

赤字プロジェクトにはしてくれるなよ。

問題を出さないようにしてくれ。

顧客からリピート注文をもらえるようにしてくれ。

ひいき顧客にしてほしい。

ほとんどの回答が から である。具体的な指示はほとんどされていない。そのような状態で、この厳しいビジネスが成功できるのだろうか？「この顧客は誰がどのような性格で、どのようなことを注意して作業せよ」「君の経験では、ここに注意して作業しないと危ないぞ」など具体的な指示はほとんどの場合されていない。失敗すると「ユーザーの要求仕様書に要求が正しく書かれていない」などと自分の責任で無いような発言を大方のベンダーの管理者がされるが、「なすべきことをしていないのではないか？」と言いたくなる。

「リピート注文がもらえればよい」との答えを最高レベルとしているベンダーもあった。前記の「お客様の6段階」を良く考えて欲しいものである。

## 4 . プロジェクト責任者の満足度の検討

### ( 1 ) ユーザー満足度調査の関係者と質問事項

プロジェクトを開発して稼動に入った後に評価を行うのは重要なことである。

今回開発したプロジェクトについて、さまざまな関係者がどのように今回のプロジェクトを評価し、改善すべきは改善し、次回のプロジェクトへ引き継ぐべきノウハウは整理して残すことは、企業の成長力を確保するためにも欠かせない。

このユーザー満足度調査について分析してみたい。

まず、この時点でのシステムユーザー（広い意味では関係者と言う表現もある）とは誰のことを指すのか？ 7種類の自社関係者と評価項目の関係分析表を下に示す。

各評価項目とその質問に答えることができる人を、主として回答可能者を、関係して答えることができる人をとして分けてみた。

B to Cのような社外の利用者や社内の業務部門の利用者には操作性、利用性を主として尋ねることになる。

利用管理者には、操作性、利用性に加えて工期・品質・費用に加えて効果やビジネスモデルの良し悪しを問うことになる。

運用管理責任者は計算機センターなどで毎日の運転を管理する上での評価をしてもらうことになる。

保守SEには、このシステムを保守する上での作業を中心とした評価を御願いすることになる。他のシステムと比較しての評価がなされる。

PM プロジェクトマネージャー、投資責任者 CIO (CEO がかねている場合もある) にはほとんど全ての質問をすることになる。

評価項目	社外利用者	社内利用者	利用管理者	運用管理責任者	保守SE	PM	CIO 投資責任者
1 . 有効利用性 (使用率、業務品質保証)							
2 . 操作性 (使用容易性)							
3 . 機能・正確性 (品質)							
4 . 保守容易性							
5 . 効果性							
6 . ビジネスモデル・業務プロセス							
7 . 工期*							
8 . 費用*							
9 . ベンダーのビジネスマナー*							
10 . 総合評価 (投資の価値)*							

(注) PM プロジェクトマネージャー

CIO チーフ・イノベーション・オフィサー

(注) \*のついた項目は、稼動直後の評価とその後の例年評価とは異なる

図表 5-1-27 評価項目の関係分析表

例えば、工期については、開発直後は開発工期に対する質問になり、稼働後の例年評価は保守依頼した案件の実行の早さを問うことになる。

費用も同じで、開発直後は開発費全体の評価となり、以降は保守・運転費用の評価となる。

ベンダーのビジネスマナーについては、開発直後はベンダーと自社社員の協調性の評価となり、以降は保守会社のSEと自社社員との協調性を問うことになる。

さて、ユーザー満足度調査となると、誰が誰に質問をするのか？が問われることになる。

質問者は社内のプロジェクト責任者であるプロジェクトマネージャーとベンダーの顧客満足度調査担当部門である。質問者、回答者と回答内容の概要を下表に整理してみた。

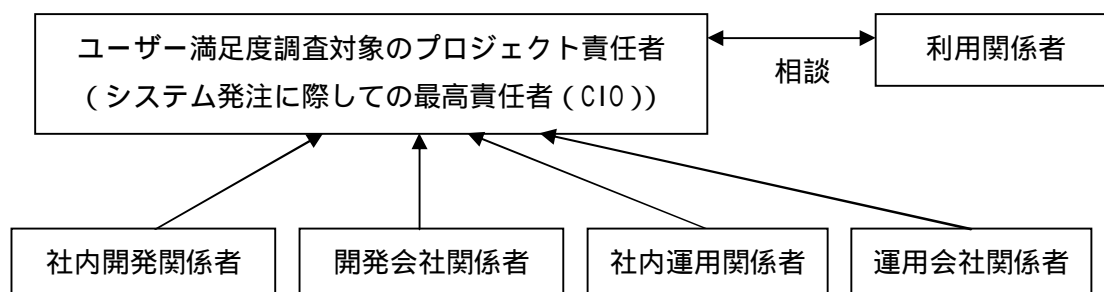
質問者 \ 回答者	データ入力者	利用部門責任者	プロジェクトマネージャー	CIO
ユーザー企業内のプロジェクトマネージャーまたはユーザー満足度調査責任者	操作性	利用性 工期・品質・費用（生産性） 投資効果		ビジネスモデル、工期・品質・費用（生産性） 投資効果 開発マナー
ベンダー		工期・品質 開発マナー	工期・品質・費用（生産性） 開発マナー 再発注是非	工期・品質・費用（生産性） 開発マナー 再発注是非

図表 5-1-28 ユーザー満足度の質問者と回答者

今回はこの中でほとんど責任を持っている、プロジェクトマネージャー、利用部門責任者、投資責任者（CIO）への質問に着目した「ユーザー満足度調査」を取り上げ検討する。

## （２）プロジェクトの評価コンセプト

### プロジェクト責任者の満足度



図表 5-1-29 プロジェクトの評価コンセプト

調査対象	調査要求者	評価ベース	調査の タイミング	
(1) 個別プロジェクトの満足度を対象	(1) 開発完了したプロジェクトの実績	開発元企業	・ USP ・ 開発結果の評価	開発完了後
(2) 一定期間のユーザー満足度を対象	(1) 顧客とあらかじめ契約を締結する場合	同上	・ SLA ・ 運用状況の評価	一定時期
	(2) 外部のリサーチ会社を実施する場合	リサーチ会社	開発 / 運用状況の評価	一定時期

注 USP: User Satisfaction Planning Sheet (開発計画合意書)

SLA: Service Level Agreement

図表 5-1-30 調査対象とタイミング

このプロジェクト責任者へのユーザー満足度調査はソフトウェアライフサイクルプロセスにおいて、図表 5-1-29 に示されるように、開発と運用の 2 面に関して問われることになる。

ケース 1

開発したプロジェクトの結果（評価ベース：USP、あるいは開発結果の評価）についての満足度を社内開発関係者がプロジェクト責任者に問う場合。

ケース 2 - 1

あらかじめ準備された SLA を基に、年度の一定時期（評価ベース：SLA、或いは運用状況の評価）に、運用会社、或いは社内運用責任者がプロジェクト責任者にユーザー満足度を問う場合。

ケース 2 - 2

SLA は準備されていないが、年度の一定時期（評価ベース：運用状況の評価）に、運用会社あるいは運用関係者がプロジェクト責任者にユーザー満足度を問う場合。

注 ( ) 内は、図表 5-1-30 の評価ベース欄にて指定した内容

### (3) プロジェクト責任者の満足度

#### a. 満足度評価の基本的考え方

満足度検討に際しては次の基本的考え方を適用する。

ユーザー満足度を構成する三要素（基本サービス、表層サービス、IT 投資価値）は全て網羅的に記述されていること。

基本サービスの評価項目はスタッフ（開発プロジェクト）或いは運用責任者（保守・運用業務）にて前提整理されること。

プロジェクト責任者は前提整理項目と代表的表層サービス項目で構成される質問事項を確認し、あくまで自分の認識と判断で評価尺度を明らかにすること。

#### b. プロジェクト責任者の満足度調査

いくつかの満足度調査表のサンプルを以下に入れる。

##### (a) ケース 1

###### 評価前提（計画 / 実績）の整理

プロジェクトの諸指標をスタッフが記入する。

プロジェクトの諸指標をスタッフが記入する。

###### ・投資費用

（ 1 . 計画以上（ % ） 2 . 計画どおり、 3 . 計画以下（ % ））

###### ・工期

（ 1 . 計画に対して遅延（ 月 ） 2 . 計画どおり、 3 . より早期立上げ（ 月 ））

###### ・品質 システムの欠陥

（ 件 / 5 百万円あるいは、FP、LOC 単位 ）は（ 1 . 計画以上、 2 . 普通、 3 . 計画以下 ）

###### ・品質 システムのレスポンス、TAT は

（ 1 . 計画以上、 2 . 普通、 3 . 計画以下 ）

###### ・効果（現在）

（ 1 . 計画以上（ % ） 2 . 計画どおり、 3 . 計画以下（ % ））

###### ・効果（一定期間経過後の予測）

（ 1 . 計画以上（ % ） 2 . 計画どおり、 3 . 計画以下（ % ））

注：効果値：実績の効果値 / 計画時の予測効果値であらわせられるもので、金額、在庫量、工期など経営戦略として指定した項目が対象

###### 質問票（評価項目と評価尺度）の検討

下記の質問票を、上記評価前提をベースにプロジェクト責任者に記入してもらおう。

質問 1：開発投資結果の全体評価（KPI（\*） 定性効果）はどのレベルでしょうか

（\*）KPI：Key Performance Indicator 重要業績達成指標

・現状（ 5 . 非常に満足、 4 . やや満足、 3 . 普通、 2 . やや不満足、 1 . 非常に不満足）

・ 2、 1 と答えられた場合はその理由を記入（ ）

・将来 ( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

質問 2 : このシステムで実現した業務プロセス、仕組みについて

( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

質問 3 : 開発に従事された方たちへの評価をお伺い致します。

( 3-1 ) ユーザーIT 関係者について

( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

( 3-2 ) ユーザー利用部門関係者について

( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

( 3-3 ) ベンダー・メンバーの働きについて

( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

( 3-4 ) 上記三者の協調体制について

( 5.非常に満足、 4.やや満足、 3.普通、 2.やや不満足、 1.非常に不満足 )

・ 2、 1 と答えられた場合はその理由を記入

( )

質問 4 : 予算はこのシステムの価値に比較してどのようにお考えでしょうか？

( 5.高い 4.やや高い 3.普通 2.やや安い 1.非常に安い )

・ 2、 1 と答えられた場合は考えられる理由を記入

( )

質問 5 : 工期についてはどのようにお考えでしょうか

( 5.非常に短い 4.やや短い 3.適当 2.やや長い 1.非常に長い )

・ 2、 1 と答えられた場合は考えられる理由を記入

( )

質問 6 : 品質についてどのようにお感じですか

( 5 . 非常に満足 4 . まあまあ満足 3 . 普通 2 . やや不満足 1 . 非常に不満足 )

・ 2、 1 と答えられた場合は考えられる理由を記入

( )

**(b) ケース2の1****前提整理**

IT 保守運用予算の計画対実績比較表、運用管理評価表（年間システムダウン回数、稼働率、回復時間（最長ダウン時間、平均リカバリー時間など）を運用会社、或いは運用関係者が準備下し、その資料を基にプロジェクト責任者が記入する。

**評価項目及び評価尺度の検討**

質問1：全体に対して以下の満足度をお答え下さい。

（各項目については運用責任者に確認します）

（5．非常に満足 4．まあまあ満足 3．普通 2．やや不満足 1．非常に不満足）

・ 2、1と答えられた場合は考えられる理由を記入

（  
）

質問2：運用費用は事業規模、他社比較、システム内容から見ていかがでしょうか

（5.安い 4.やや安い 3.普通 2.少し高め 1.高い）

・ 2、1と答えられた場合は考えられる理由，期待する対策などを記入

（  
）

質問3：利用状況には満足されておられますか

（5．非常に満足 4．まあまあ満足 3．普通 2．やや不満足 1．非常に不満足）

・ 2、1と答えられた場合は考えられる理由を記入

（  
）

## 5 . ユーザー満足度調査の計画と推進

### ( 1 ) 開発計画合意書 ( USP ) の作成

(USP : User Satisfaction Planning sheet)

当プロジェクトが完了した時 (あるいは契約フェーズが完了した時に) 、プロジェクト責任者が、何をどのような状態にさせて欲しいと願っているか、つまり IT 投資の評価項目をあらかじめ明らかにし、ユーザー満足度を評価する基本資料として使用する。

#### a . 利用者の満足度

##### ( a ) 基本サービス

品質 : 納入物に対する評価項目と評価尺度 (品質目標値) を明確にする。

- ・対象とするドキュメント : 納入検査に持ち込んだ納入物
- ・システムの欠陥性、システムの性能性、システムの信頼性を評価項目とし、それぞれに対する目標値を設定する。
- ・評価時期 : 納入テスト ~ フォロー間。

納期 / 工期 : 成果物を規定し何時までに何を完了させるのかを明確にする。

費用 : 見積前提 / 条件に留意し、契約金額 + 追加金額を明確にする。

##### ( b ) 表層サービス

適用する評価項目および評価尺度を明確にする。

基本サービス、表層サービスともベンダー側の一方的義務ではなく、ユーザー側の資料提供、説明、設計資料などの検査の精度によるところが大きいので、あくまでも計画値であって契約保証値ではない。両者の役割を明確化したうえで、相互努力目標値である。

#### b . プロジェクト責任者の満足度

効果 : 時系列の変化値に留意した CIO の期待値を明確にさせる。

- ・数値化できるもの、数値化しにくいものの両者について指標を確認しておくこと。
- ・前提整理方法、測定方法、確認方法、評価方法など手続きについても規定しておくこと。

目的を達成するための仕組み : 新ビジネスモデルについて納得の程度は ?

プロジェクト推進上の課題 : IT 担当部門とベンダーとの共同推進作業などの満足度

#### c . 計画 ~ 実績対比時の留意点

開発完了時あるいは契約フェーズ別完了時に、この USP と実績を対比し、

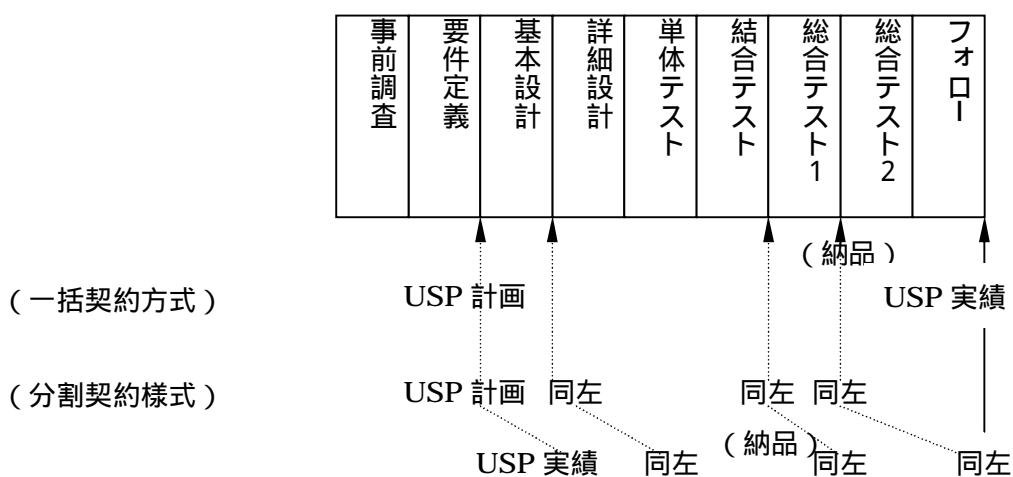
- ・いかなる対策をとればよかったのか
- ・今後の前向きなアクションの可能性

などをベンダー・ユーザー双方で確認する。

本番後の USP 実績評価の実施時期もフォロー期間を予測して計画時に設定すること。

実際のシステムの効果を確認した後でないとは評価できない項目は、改めて効果確認の時期を設定し見直しをすること。

## d . 契約フェーズと USP の関連



注

- ・分割契約時には総合テスト1のフェーズはない。
- ・定期的を実施するユーザー満足度調査とは、別なものと考えること。

図表 5-1-31 契約フェーズと USP の関連

既に、ここまでに開発・保守・運用についての業務、契約の多様性、関係者の輻輳性、評価項目の複雑性等について見てきたが、できるだけ簡潔に日本型開発評価尺度を調査する技法をまとめて見た。日本の情報化社会からトラブルを無くすため、あるいは日本のソフトウェア開発及びシステムの運用力を高めるための、実態を確認、追及するための技法である。



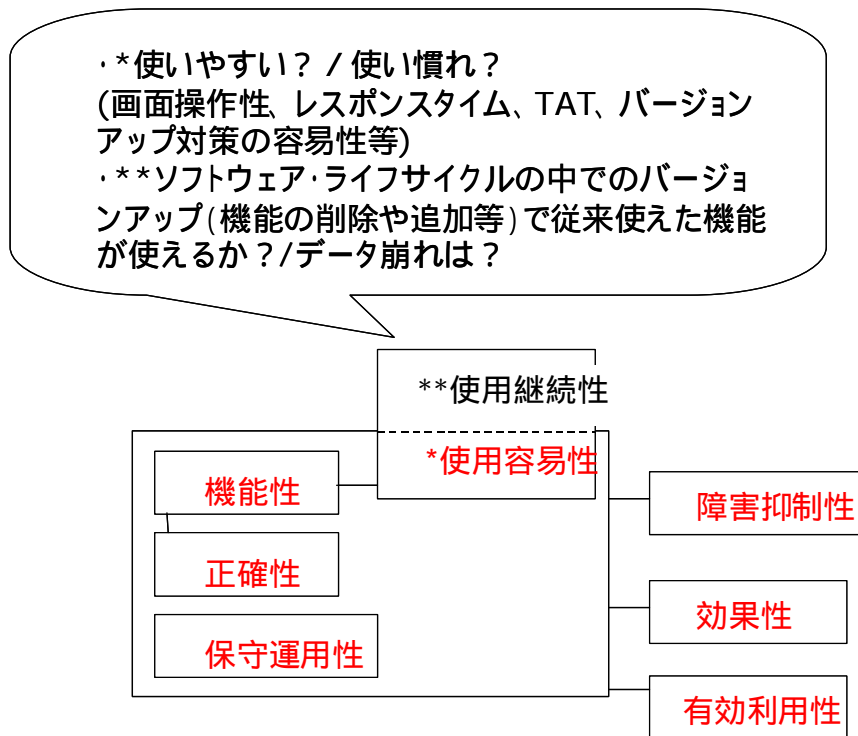
## 第5章 利用評価

### 第2節 ユーザビリティ

---

1. ユーザーの望む品質特性とは何か
2. ISOで定めるユーザビリティ (ISO13407)
3. 見込み生産商品とアプリケーションシステムのユーザー満足度対策の差
4. ユーザビリティとプロジェクトマネジメントとの関係
5. ユーザビリティ検証
6. 単体テスト完了時
7. 効果
8. 参考：ユーザビリティチェックリスト

## 1 . ユーザーの望む品質特性とは何か



図表 5-2-1 ユーザーの望む品質特性とは

最初にシステム開発プロジェクトを対象にして「ユーザビリティ・使用容易性」が含まれる「ユーザーの望む品質コンセプトとは何か?」を、整理してみる。JUAS で議論してきた、ユーザーの望む品質コンセプトは上図のごとく

「まず良い機能がついていて、正しく動き、使いやすく、システムの保守・運用もしやすいこと」が基本となる。各要素を以下に解説する。

**機能性** 良い機能とは、利用者の必要とする、優れた機能である。

**正確性** 正しく動くかどうかは、やや複雑になる。欠陥の全くないソフトウェアを最初から作成するのはかなり至難の技である。ユーザーは「ベンダーが納入してきてから、安定稼働にいたるまでに一定の欠陥数に収めること」をベンダーに望んでいる。

ともすれば、ソフトウェア開発・運用の世界では、このような目標値がなく常識が通じがたい商品が横行することになる。JUAS はその点について、「品質・工期・生産性の目標値を持つこと」を提言し、標準値を提供し、検証している。

さて次の「使用容易性はどのように取り扱えばよいのか?」

日常私たちが扱うデータの操作は入力画面を通じて行うことが殆どであるが、最近は Web 技術が向上し複雑な仕組みも準備することができるようになって操作性の向上が実現できるようになっている。

しかし、一度も画面操作の教育を受けることもない一般消費者が、インターネットの画面を使って購入注文情報を入力しようとしても、なかなか受け付けてもらえず、途中で「もうやめた」と投げ出すケースが多い。

「社長、貴方は自社の画面から、貴方の会社に対して注文情報を入力することが出来ますか？」と問いただしたくなる。きっと「もっと売れても良いのに、B to C は思ったほど普及してこない」などとつぶやいている社長もおられるだろうが、実は発注者の入力作業の困難性が原因だとは感じていないのではないか？

企業内の従業員を対象にしたシステムも「分かりにくい、使いにくい、面倒だ」などの理由で「作成したのに使ってもらえない」と頭を抱えている SE や管理者も結構多い。

一方最近のシステム開発工期を調査したところ、設計期間と開発構築期間とテスト期間の比率は「3 : 3 : 4」となっていることが分かった。

また日本で最近開発したシステムの工期全体も Boem 先生の唱えられている COCOMO 法に則り、 $2.7 \times (\text{投入人月の立方根})$  となっており、工期短縮がさげばれている割には「工期は短くなっていない」。つまり最後のテスト工期を長く確保する開発方式になっている。

このテスト工期の中に「操作性が悪い」「使いにくいので、使いやすいように画面を修正して欲しい」との利用者の要求に応じて画面を修正する作業が含まれている。

このような操作性に関係した問題を、どのようにエンジニアリングし、乗り越えて行けばよいのか？

開発工程の最後にいたってから、修正や再作成を要求されないように「使用容易性」を取り込んだシステムエンジニアリングが今望まれている。

・JUAS ではこのような問題解決を図るためにユーザビリティプロジェクトチームを 2004 年度に編成した。2005 年度も継続研究中である

- ・企業の基幹業務システムについてのユーザビリティ研究は珍しいこと、
- ・システム開発のプロセスとユーザビリティの関係を追及することも珍しいこと
- ・企業内のシステムエンジニアが企業の枠を超えて、この問題について議論することも珍しいこと

などの特徴を備えている。

なお、難問を短時間で解くために、この道の日本の第一人者である文部科学省メディア教育開発センター黒須教授に指導をお願いし、13 名の企業代表メンバーのチームを編成した。

なお、このプロジェクトは日本小型自動車振興会の補助金を頂き報告書を作成した。

## 2 . ISO で定めるユーザビリティ ( ISO 13407 )

ユーザビリティを学ぶ者が、最初に目にするのが、この ISO13407 である。

ユーザビリティの定義、考え方の基本をここから学ぶことができる。

### ( 1 ) ISO 13407 の概要

ISO13407 は正式なタイトルを”Human-centered Design Processes for Interactive System” (インタラクティブ (対話型) システムの人間中心設計プロセス) といい、1999 年 6 月に発行した人間工学分野の国際規格である。設計プロセスを人間中心設計にすることで、製品のユーザビリティを高めようという考え方である。

ISO13407 では、ユーザビリティという概念は一口に言って「使いやすさ」あるいは「使い勝手」ということであるが、以下の 3 つの要素から構成されると定義している。

有効さ ( effectiveness ): ユーザーが指定された目標を達成する上での正確さ、完全性

効率 (efficiency) : ユーザーが目標を達成する際に、正確さと完全性に費やした資源

満足度 (satisfaction) : 製品を使用する際の、不快感のなさ、及び肯定的な態度

このユーザビリティ概念は通常大きく 3 つの異なった意味で使用されており、一番限定した意味で使われているのは、「バリアフリー」といった障害者への対応、次にやや広い意味では機能や性能のネガティブな側面の改善といった製品の「ユーティリティ」という概念が存在していたが、必ずしもそれだけではユーザーの生活や仕事に最適な製品の開発には十分つながらなかった。そして ISO13407 により、「使いやすさ」あるいは「使い勝手」という意味で一番広い概念となった。

従って、従来の人間工学の規格、代表的なものとしては身体計測学的もしくは生理学的な基準を定めた規格ではなく、「製品には利用品質 (Quality in Use) が必要である」という主張のもと、この品質を製品設計の中で確保するための原理とその原理のもとでの設計プロセスの導入・管理方法を規定したものであり、人間工学分野でのプロセス規格の先駆けの 1 つとして位置付けられている。したがって ISO13407 には、数値的な規定は一切含まれていない。

また、ISO13407 は、コンピュータを用いたインタラクティブシステム (ハードおよびソフトの両方を含む) のライフサイクルに対して、人間中心設計プロセスを確立するための指針を与える規格であり、特徴としてまず「適用範囲が広い」ということであり、もう 1 つは「設計プロセスを対象とする」ということである。

#### < ISO13407 の歴史 >

ではこのような人間中心設計の考え方がどのように醸成されてきたか沿革について触れてみたいが、背景としてヨーロッパ、特にドイツ、イギリス、スウェーデンでは、人間中心 (Human-centered) という生活の質 (quality of life) を重視する考え方があり、1970 年代から芽生え、1980 年代には国際協調に基づく戦略的な研究や人間中心設計提案等に発展し、1996 年以降 ISO9241-11、13407 への取り組みやユーザビリティ支援のネットワーク化等となった。このようなヨーロッパにおける人間中心設計のための研究開発の成果が反映されたのには、『情報技術に対する人間工学 (ITE : Information Technology Ergonomics)』と呼ばれた研究領域が、背景にあった。

以上が、EU における人間中心設計の背景であるが、注目すべきは、その戦略性である。20 年以上前から、IT には、ユーザビリティを含めたヒューマンファクターの問題が不可欠であるという共通認識があり、産業界に対しては、人間中心設計を手法や技法の域にとどめず、システム開発の主流に位置付けるように発展してきたことは見逃せない点である。これに関連し、ソフトウェア工学やシステム工学と融合した人間工学領域も生まれている。ユーザビリティ工学、要求定義工学が典型的なものである。

#### < 日本における人間中心設計への取り組み >

EU では、個人や社会へ IT が与える影響を早い時期から認識し、包括的な開発方法論の確立を思考してきたが、わが国の開発体制の現状を見ると、多くの場合、製品の機能、性能、デザインの開発を推進する仕組みにはなっているものの、ユーザ視点のコントロールが明確には組み込まれていない。

各企業によって、人間工学の貢献の度合いはバラバラであったが、総体的には特定の開発プロセスに限定されているだけで、開発ライフサイクル全体には大きく貢献しているとはいえなかった。ライフサイクルを開発・設計・評価・設置と分けた場合、評価段階で利用される場合が比較的多いようであった。企画段階では、ユー

ニーズの把握、競合分析はおこなわれるものの、この結果がユーザー関連の問題として一貫して生かされることは稀であり、設置やそれ以降のサービスに関しては対応外となっていた。

このように部分的あるいは補助的であるために、一貫した活動となっておらず、ユーザーと製品の不整合からくる様々なリスクを背負っていることになる。このリスクには、例えば、顧客満足、信頼性の低下から商品のブランド力を減少させ、売上が減少する事態を生じさせることなど、多くを指摘できる。

## (2) 人間中心設計の考え方とその効果

人間中心設計のベースとなる基本コンセプトは、『利用品質』と『プロセスマネジメント』である。『利用者から見た製品の品質』即ち『利用品質』の改善と保証のメカニズムを製品製造サイドの経営システムの中に組み込むことにある。この体制を構築することは、結果として、ユーザーや顧客との関係を深めて、顧客価値を創造するための安定した基盤を形成することにつながっていくことになる。

<人間中心設計の基本コンセプト>

利用品質マネジメントを行うシステム

人間中心設計プロセスマネジメント

人間中心設計プロセスを推進する経営システム

ISO13407を導入する効果を、ユーザーとメーカーともにWIN-WINの関係が成り立つことが説明されている。即ち、ユーザー側は、その製品を満足して使うことができ、生産性が上がること、メーカー側は、ユーザーサポートに関するコストを減らし、市場で使いやすさを際立たせることが出来るとしている。

具体的に企業にとっての効果としては次のようなことがあげられる。

- 企画段階から、次の設計段階以降へのフィードフォワードが発生する。例えば設計仕様やUI (User Interface) 仕様の提案、製品評価計画、販売やサービス戦略への提案等々である。それは、人間中心設計が製品開発ライフサイクル全体に関与するためである。即ち、開発の前倒しが起こり、コストダウンにつながる。
- ユーザー視点が浸透、定着することによって、企画提案力を向上させる知識基盤を形成する。同時に、提案した製品の受け入れが安定してくる。市場からの撤退のような大きな失敗が軽減される。
- モノとしての製品からサービス、ソリューション提供へのシフトが容易になる。なぜなら、サービス、ソリューションを提供するためには、ユーザーの利用状況へのより深い理解が必要となるためである。
- 製品のユーザーへ及ぼす危険性等のリスクを軽減させ、PL (Product Liability) 対策を推進できる。

以上が、開発ライフサイクル上で期待される効果である。

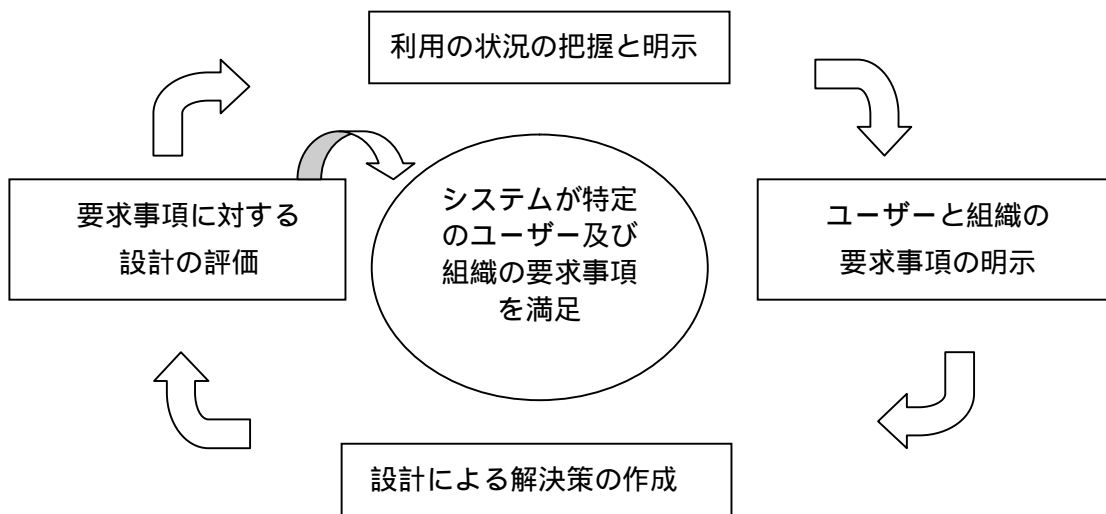
人間中心設計が企業に制度化されることによる波及的な効果を考えると、ユーザー指向の企業が認知され、製品のブランド力が高まり、価格プレミアムに繋がることも期待できる。また、顧客中心の経営システムを実現化させることにもつながり、経営品質の向上にもつながる。顧客中心主義は、現代経営における基本軸であり、その実体化には人間中心設計が有効な参照モデルとなり得る。

### (3) 人間中心設計のプロセス

ISO13407 では人間中心設計プロセスとして次の4つのプロセスを定義している。

- 利用の状況の把握と明示
- ユーザーと組織の要求事項の明示
- 設計による解決策の作成
- 要求事項に対する設計の評価

この4つのプロセスを目標達成できるまで繰り返し行うことにより、人間中心設計が確保されるとしている。



図表 5-2-2 ISO13407 の4つのプロセス

このプロセスの組み立て方は、製品の種類、製品のライフサイクル、企業の組織・文化に依存するものであり、この依存の度合いによって多様な形態があり得る。従って、ISO13407 の導入を検討する企業は、その企業の現状を十分に把握した上で、自社に合った形で4つのプロセスを組み入れることが求められる。

次に、これら4つのプロセスにどのような活動が求められるか見ていきたい。

企業内のシステムを開発する場合にも、この図は意味を持っている。

「ユーザーと組織の要求事項の把握」はシステム開発をする場合の最初に確認しなければならない重要事項である。

使用容易性以前に

- 「このシステムは何を目的にしてつくるのか？」
- 「手作業をシステムに委ねる必要が本当にあるのか？」
- 「効果と IT 投資はバランスするのか？」
- 「開発したが使われない可能性はないのか？」
- 「システムの寿命はどの程度なのか？」
- 「環境の変化に耐えうるのか？」
- 「情報共有とどのような関係にあるのか？」

などの基本的事項を確認してから開発に着手せねばならない。

つまりシステム設計の良し悪し以前に、組織として確認すべきことがあることを意味している。通常の企業でのシステム開発では「システムレビュー」「あるいは「システム企画会議」などの中でこの作業は行われている事に着目して欲しい。

#### a . 利用の状況の把握と明示

利用の状況(Context of Use )とは、ユーザーがその製品を使う状況であり、大きく分けて次の3つに分けられる。

- ・ユーザーの特性：知識、スキル、経験等想定ユーザーの特性
  - ・仕事の特性：ユーザーが製品を使用して実行する仕事の特性
  - ・環境の特性：物理的環境（空間、温熱、証明等）及び社会的環境（法律、文化等）の特性
- 製品規格を行うにあたっては、市場調査、マーケティング、技術調査等により、様々な情報が収集・分析されるが、その過程で「この製品は、どんな人が、どのように、どんな環境の中で、使用するのか」を明確にする諸活動の総体が、このプロセスである。

#### b . ユーザーと組織の要求事項の明示

人間中心設計では、従来行われてきた製品への要求仕様の記述にあたって、ユーザーの要求を「利用の状況」との関係から明示的に記述することが求められる。記述する項目としては、例えば次のようなものがある。

- ・ユーザーの範囲の記述
- ・明確な設計目標
- ・異なった要求項目の優先順位
- ・設計評価のための測定可能な基準
- ・法的要求項目

要求仕様を作成する活動は、人間中心設計に限らず通常実施されているが、人間中心設計においては、前項の「利用の状況」の把握に関連してユーザーとそれを取り巻く組織の要求事項を特定・記述することが必要とされている。尚、ここでいう要求仕様とは、設計のための要件定義とはことなり、あくまでユーザーの立場に立った要求仕様である。

#### c . 設計による解決策の作成

このプロセスは、b . で定まった要求仕様に従って具体的な設計を行い、設計の妥当性を確認するプロセスである。手順としては以下ようになる。

- ・多様な職種に基づいた設計を実現するための既存の知識（人間工学、認知科学等々）の活用
- ・シミュレーション、モデル、モックアップ等を用いた具体的な設計
- ・プロトタイプのユーザーへの提供と模擬タスクの実行

これらの手順を目標が達成できるまで繰り返すことにより、人間中心設計の具体化を図ることになる。尚、ここでのプロトタイプは、スケッチのような簡単なものからシミュレーションまでを含むものである。

#### d . 要求事項に対する設計の評価

設計評価は、人間中心設計の中で重要なステップであり、

- ・ 設計改善のためのフィードバックの提供
- ・ ユーザー及び組織の目標を達成したかどうかの査定
- ・ 長期的なモニタリング

を行う。

ユーザビリティ評価における、専門家評価、ユーザテスト等が実施されるのは、このステップにおいてである。ISO13407 では、ユーザテストに対する要求（推奨）として次のような項目が挙げられている。

- ・ 十分なユーザー数を確保すること
- ・ 利用の状況のプロファイルに沿ったユーザーを被験者に選定すること
- ・ 全ての設計目標をテストすること
- ・ テスト手法、データ収集方法として妥当なものをを用いること
- ・ テスト条件を文書化して記録すること

尚、ユーザーの健康及び安全を考慮すると、製品及びシステムの長期間の使用の中で問題点が明らかになる場合がある。このステップに「長期的なモニタリング」が含まれているのはこのためである。但し、このモニタリングは製品出荷後に実施されてもよい。

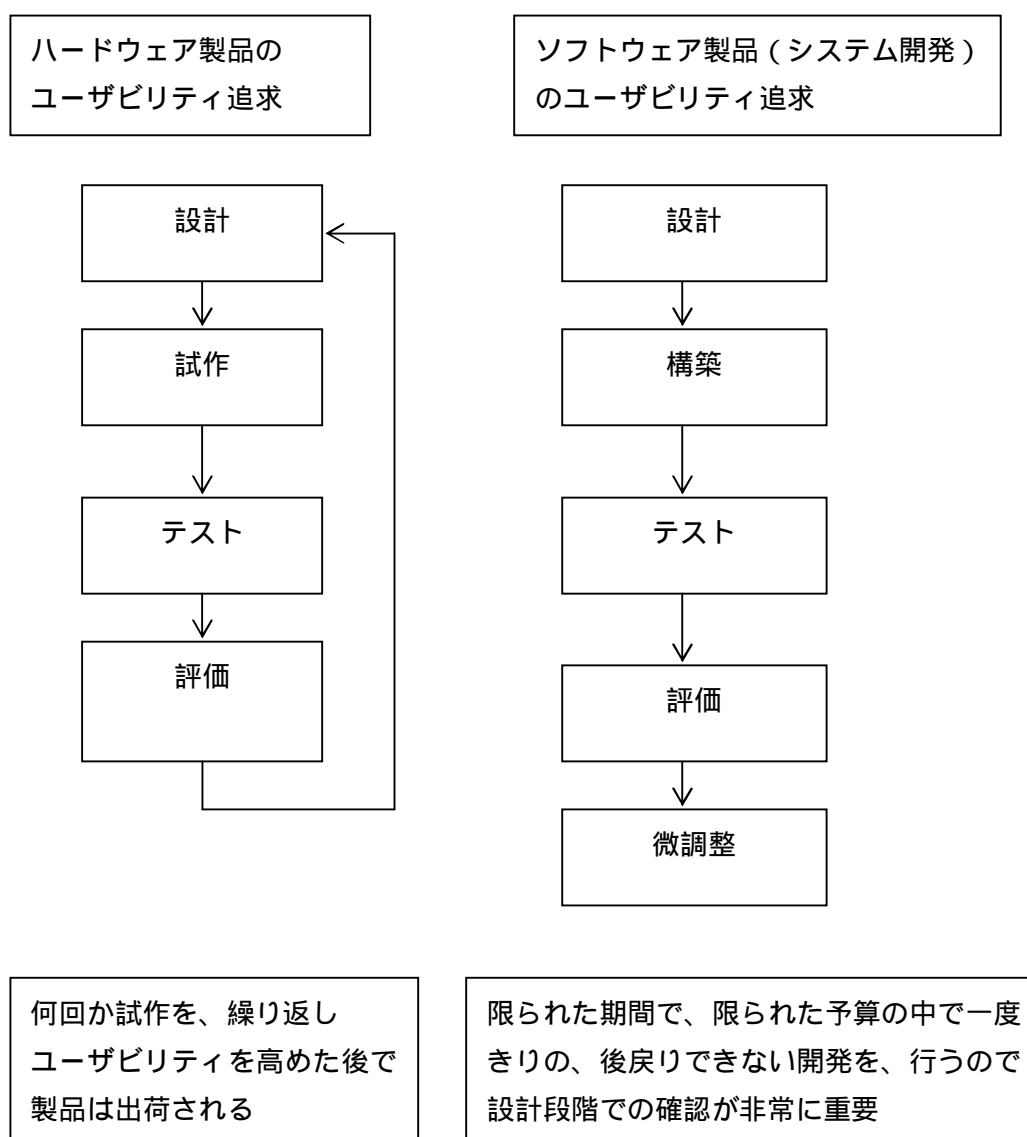
#### e . 適合条件

ISO13407 は「規格」である以上、その規格に適合しているかどうかを判断する基準がなければならぬ。しかし、これまで示してきたように、ISO13407 は、設計の考え方と手順を規定したものであるため、数値的な判断基準を設定することができない。これに代わって設定されているものが「文書化」という要求である。ISO13407 で想定している文書は次の 6 通りのものである。

- ・ 設計計画および管理書
- ・ 利用の状況
- ・ 要求仕様書
- ・ プロトタイプ
- ・ 評価報告書
- ・ 人間中心設計報告書

最後のものは、全体のまとめの位置付けである。また、プロトタイプは、「文書」である必要はない。ISO13407 では、文書を残すことを要求しているが、その内容・レベルについては規定しておらず、関係者間の話し合いで決めることとしている。これは規格の適合条件としては、非常に「ルーズ」なものとも受け取られるが、「関係者間の合意が取れない限り適合したものとみなさない」と読めば、非常に厳しい条件ともいえる。

## 3 . ユーザビリティ対策の差



図表 5-2-3 ユーザビリティ対策の差

見込み商品は、設計 試作 テスト ユーザビリティの確認のサイクルを数回にわたって繰り返し、問題がないことを確認してから、出荷するのが通常のステップである。

しかし企業の基幹業務システムなどのアプリケーションシステムを、ウォーターフォール法を用いて開発する場合は

設計 構築 テストを1サイクルで行うのが通常であり、工期も費用も少ない中でのユーザビリティ追求となる。後戻りすることは、費用増大、工期遅延を招くので、ほとんどの場合に、操作性の向上のための抜本の変更は出来ない。

イテレーション開発（反復開発）の場合は開発工程の繰り返しは、可能ではあるが、実際問題として、やはり工期、費用の制約から、通常なら2～3回の繰り返ししか出来ない。

結局ユーザビリティを高めるためには

「基本設計の中の1要素としてユーザビリティを取り込むこと」と

「単体テスト完了時点でユーザーに結果を披露し微調整を行う」

の2項目を計画的に推進することが開発のポイントとなる。

JUASの2004年度のソフトウェアメトリックス調査によると、システム開発の工期は、設計・構築・テストの期間比は、3:3:4で結合テスト、総合テスト(ベンダー内での総合テスト+顧客内総合テスト)のテスト期間が意外にも長いことが判明した。

これは最近、システムが少しでもトラブルが発生するとマスコミなどに騒がれるばかりでなく、利用者や顧客に迷惑をかけるので、これを避けるために最後の確認を重視していることも影響している。

特に顧客内総合テスト期間が長いのは、最後の総合テスト期間でさまざまな確認が行われるためである。

仕様上の確認

あらかじめどのような操作性になるのか?を確認しないまま発注してしまうと、本番間近になって利用者に新システムの公開をしたところ「このような操作性ではとても簡単に入力できない。今までよりも入力時間がかかっても良いですか?」などとのクレームを受けその対応に追われることになる。

利用者への教育が行われる。

開発したシステムを移管し、運用管理する計算センターのオペレーション部隊のための準備に時間がかかる。

「開発はしたが運転管理できるレベルの準備が行われていない。これではシステムオペレーションは引き受けられない。維持運用管理できるように修正してください」と言われることがある。そのための準備をこの総合テスト期間で実施するので、この期間が長くなることがある。

単体テストが不十分なために欠陥が見つかりこの期間で修正することがある

この4項目が結合テスト～総合テスト期間の主要業務である。

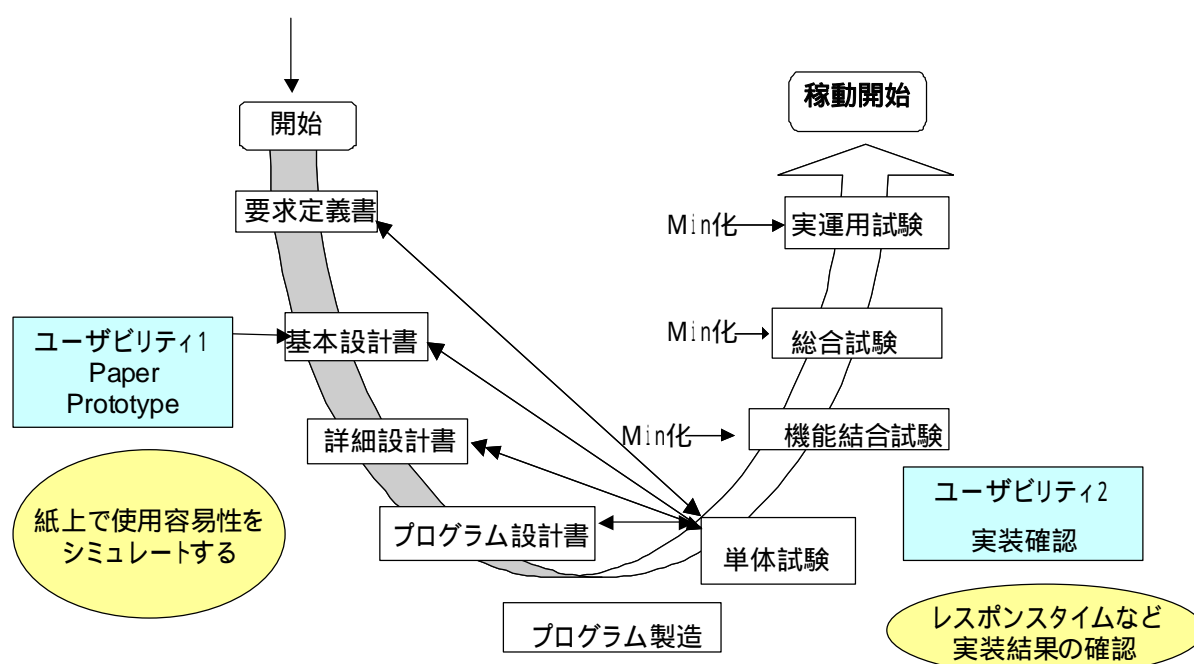
は別として、 、 、 の3項目を基本設計の段階から組み込むことの意義は大きい。

## 4 . ユーザビリティとプロジェクトマネジメントとの関係

JUAS の調査結果によれば、プロジェクトマネジメントにおける手法は、ウォーターフォール型が 90% 以上を占め、反復型は非常に少ない。これは基幹業務システムなどの比較的大きなシステム開発は、サブシステム相互間の調整が必要であり、こまめに仕様が修正できる反復型よりも確実にシステム開発を行いやすい点が評価されているためと思われる。

もっともウォーターフォール型も単体テスト後に修正を一部取り込むことになるので、2 回は仕様を見直すことになり、2 回反復型（イテレーション型）とも言える。

逆に言えば、反復型（イテレーション型）開発も、ウォーターフォール型の繰り返しとなる。



図表 5-2-4 U字型開発でのユーザビリティ

このU字型開発手法はJUASが、基幹業務プロジェクトなどを円滑に開発するために、日本の開発体制にふさわしい開発手法として編み出したものである。

大型プロジェクトにおいて、欧米の開発のように、開発要員を全て手元において開発するケースは少ない。特にプログラム作成部分は、外部の専門企業に委託するケースが多い。

そのような開発体制の場合は、単体テストが完了次第、外部の専門企業のプログラマーはリリースされる。したがって総合テストの時期にはプログラムを書いた人は近くにいないことが多い。

そのような開発体制でも高品質のシステムを安価に作成するためには、単体テストが完了した場合には、ほとんどシステムは出来上がっているこのU字型システム開発方法を推進する必要がある。詳しくは、第3章第5節を参照いただきたいが、要点を個々に集約してみる。

### ( 1 ) 要求仕様の作成

良質・安価・短工期のシステム開発を実施するためには、まず利用者自身が「何をしたいのか？」を要求仕様書として作成する必要がある。

利用者であるから SE のような専門知識を持ち合わせないが、以下の項目の定義は可能である。

何をしたいのか

そのためにはどのような機能が必要なのか？を記述する。

画面設計も粗くて良いので作成する。画面には入力項目が載っているが、各項目には当然「新規・訂正・追加、削除」などの機能は必要である。これらの機能の設計は自社あるいはベンダーの SE に任せる。必要機能は例外処理も含めて、構造図程度でも良いので、もれなく記載する。

「何をして欲しいのか？」を書くが、どのような手順でプログラムを書くか？は、これも自社あるいはベンダーの SE に任せる。例外事項でシステム化を望む範囲は漏れなく例外作業項目を記載する。

整理しきれない例外事項は異常作業としてシステム外扱いとする。

この仕事の割り切りをしないと、ユーザーの協力は得にくい。

この要求仕様書を元に自社あるいはベンダーの SE が、プログラムに繋がるように処理設計を行う。

### ( 2 ) 設計書のレビュー

自社あるいはベンダーの SE が作成した仕様書のチェックは、作成責任組織でも当然行うが、発注者サイドのチェックも欠かせない。このレビュー時間は作成時間のおおよそ 10% をかけて行う。

この程度の時間をかけないと修正箇所は見つからない。

とくに曖昧な表現の形容詞、副詞の利用には注意を要する。

名詞は単数が、複数かに気を配ってレビューする。

### ( 3 ) 単体テスト時にはコンバージョンデータを用意

全く新しいシステムは少なく、置き換え再構築のシステムが多い。

この場合は現在使用しているデータをテストに活用することが出来る。

総合テストに間に合うように既存データのコンバージョンをするのではなく、単体テストに間に合うようにコンバージョンをし、単体テストからこのデータの活用を図る。

### ( 4 ) 基本設計段階はユーザビリティ確認のプロセスを入れよ

この内容は次の章に示している。

## 5 . ユーザビリティ検証

### ( 1 ) ユーザビリティの検討開始

プロジェクト開発のフェーズの中で、基本設計と単体テスト後の2箇所で以下の要領でユーザビリティを迫及する。企業のビジネスシステムにおけるユーザビリティ検討手法は、各種試みがなされているが、今後の研究に待つところが多い。

しかし、重要なポイントは二つある

ユーザビリティのノウハウを各企業内に蓄積するために、ユーザビリティの担当者を決めること

一人のキーマンにノウハウを集約し整理蓄積することが肝心である。

全プロジェクトのユーザビリティのチェックにこのキーマンを参画させることが、早期蓄積のポイントとなる。

活動を始めること

企業情報システム開発についてのユーザビリティ手法は、定型化されたとは言えないものの、チェックの要領は、この章に記述されてある。

チームを作ってまず検討を開始してみることが、次のポイントとなる。

各人の感性はどのように発揮されるのか？

各人の気配りの差がどの程度あるのか？

などを感じながら、このユーザビリティチェック・プロセスを取り入れることに意義がある。

### ( 2 ) チームの結成

参加者：以下のメンバーで構成される

チームリーダー

ユーザビリティのキーマン

利用者代表 入力操作を実施する人、データ管理の責任者（入力操作者の上司）

システム開発担当 SE、システム保守担当者

チーム：上記チームを可能ならば、3チーム程度準備するのが良い。

人が変われば、見方が変わることが分かる

小企業の場合、あるいは利用者が少ない場合は1チームでもかまわない

### ( 3 ) 実施時期 1 基本設計段階

基本設計段階で、ある程度画面・帳票の姿が見えてきた時期

### ( 4 ) 準備

作成しようとするシステムの目的、効果の実現方法、作業改善・変更のポイント、業務の流れ図、企業内関連システムなどを記載したドキュメントを準備し、参加者の基本認識を徹底させること。

## (5) 実施方法

### a. 紙上検討法 (ペーパープロトタイピング)

#### (a) ペーパープロトタイピングとは何か?

ペーパープロトタイピングとはユーザビリティテストの一種である。

新しく作成しようとしている画面を手書きで紙の上に書き、ユーザーが入力操作を紙の上で行う。そのアクションに基づき、あたかもコンピュータで実行しているように、コンピュータ役が紙切れを動かしてインターフェースの動作をシミュレートする。

ただしインターフェースの動作に関しては、ユーザーには何も説明はしない。進行役がテストを進行させ観察者が状況を観察し記録する。

ユーザーがこの操作で困っていることはないか?

システムから出されるアクションのガイドやメッセージは妥当か?

以上、処理は簡単なのか?

オペレーションで考え込むことはないか?などを観察者は見抜きメモをしておく。

#### (b) ペーパープロトタイピングのメリット

紙の上での操作ですむので実装上の時間が省略され、利用者が目の前にいるうちに修正結果がフィードバックできる。(改良は消しゴムで消し修正するだけですむ)

開発プロセスの早い段階で利用者の反応を確認できる

数多くのアイデアを試すことが出来る。

開発チーム内や開発チームと利用者とのコミュニケーションが活性化される

技術的なスキルを必要としないのでさまざまなチームが協力しあえる

製品開発プロセスにおいて開発者に創造的な思考を促す

早く結論が出る上に低コストである

#### (c) 役割

ユーザー 実際の利用者に操作してもらう

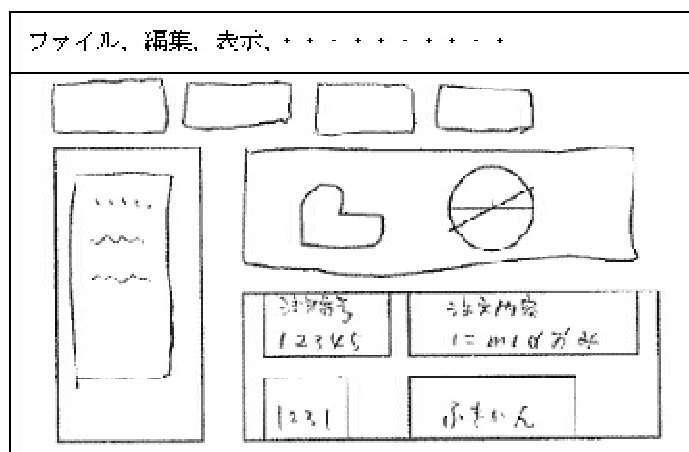
コンピュータ役

入力に対してあたかもコンピュータが動いたかのように反応を想定し紙を動かす

進行役 作業の継続を促す

観察者 状況、課題を記録する

## (d) サンプル



図表 5-2-5 ペーパープロトタイピングのサンプル

## (e) 参考：使用する道具

道具	用途	備考
ホワイトボード	ペーパープロトタイプ要素を配置するための背景として使用	30×40cm程度
白い紙	大き目のプロトタイプ要素を描いたりメモをとったりするために使用	
無地のインデックスカード	小さめのプロトタイプ部品	15×24cmと12×18cm
マーカー、ペン	プロトタイプを手書きするのに使用	黒およびカラー
蛍光ペン	透明フィルム、はがせるテープと組み合わせて要素を強調表示する	
透明粘着テープ、貼ってはがせるのり、付箋紙	プロトタイプ部品を固定するのに使用	
透明フィルム	プロトタイプの上に重ねて、この上にデータを手書きする	アセテート製
水性OHP用マーカー	透明フィルム上に書き込み用	消すのはぬれたタオル
修正液	プロトタイプに小さな変更をするのに使用	
スチレンボード	3次元表示のプロトタイプ作成に使用	Fome-Cor

注：模造紙は大きすぎる、定規は出来る限り使用しない

図表 5-2-6 ペーパープロトタイピング（使用する道具）

## b. シミュレートモデル法

本物とはやや異なるが、画面を端末上に配置し、入力項目を擬似入力する仕組みを既製品で入手することが出来る。そのモデルを活用して操作性の是非を検討すること

（チェック項目は上記と同じであるが、ツールの良い悪しにかかわるような瑣末な検討に陥らないように配慮すること）

c . マイクロシナリオ方式

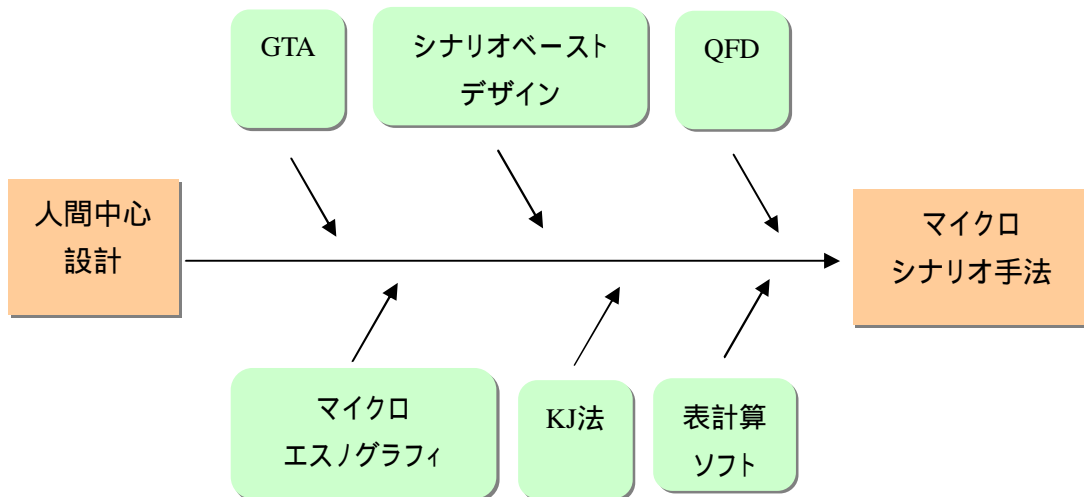
マイクロシナリオ手法の特徴

マイクロシナリオ手法、すなわちインタビューから要求仕様を構築するこの手法の特徴を要約すると以下ようになる。

- ・ライフスタイル、つまり生活実態を理解することは、新しい技術の方向性、その利用のしかたを考える上で基本となるアプローチである。こうしたアプローチは「人間中心設計」と呼ばれている。ISO13407 という規格によって、そのアプローチが明確化された人間中心設計の考え方では、四つのプロセスにもとづいて人間中心設計を達成することが主張されている。その第一プロセスがユーザーの理解、つまり「利用状況の明確化」である。
- ・近年、その第一プロセスに対する関心が高まってきている。ユーザーのニーズや必要性に適合した人工物を設計すれば、それは「売れる」ものである、というように理解されるようになってきたからと考えられる。
- ・この手法では、従来エスノグラフィーや社会学で利用されてきた質的手法(インタビューなど)を出発点とし、それにこれまで開発されてきた様々な手法を統合することにより、あたらしい要求仕様構築手法としている。

すなわち、

- ・ライフスタイルに関するインタビュー調査にもとづいて、問題点を抽出し、それを解決するための要求仕様を構築する手法である。
- ・表形式でデータを扱うため、問題点が整理しやすく、ロジックを確認しながら要求仕様をまとめることができる。



図表 5-2-7 マイクロシナリオ手法

注 1 : GTA Grounded Theory Approach

対象となるシステムを操作する人についての理解を深めるための分析法

注 2 : QFD Quality Function Deployment 品質機能の加重配置

ユーザビリティを確保するためのアクションを選択する必要がある場合に、技術の適用可能性や コストなどの重要度に応じて重み付けをし、重みとアクションにつけられたポイントの積和で順位をつける方法。

注3：KJ法 川喜田二郎法

思いついたアイデアをカードにして並べ、それらのグルーピングを考え、全体を関連づけてベストアイデアへ収束させる発想整理法

注4：マイクロエスノグラフィ

あるアクションに反応して人はどのような行動をとるか？を観察しその背景を含めて理解し記録する方法。

## 問題マイクロシナリオ(sMS)の作成

### 1.問題マイクロシナリオの構成

問題マイクロシナリオを作成する際は、一人の情報提供者から得たインタビューデータについて、一つの共通情報(GI: Ground Information)と複数の問題マイクロシナリオを作成する。

### 2.共通情報の記述

共通情報というのは、その情報提供者についての一般的な情報のことで、ライフヒストリー、現在の生活状況や生活環境、ものごとの考え方、身体的特徴、など、問題マイクロシナリオに共通する情報のことをいう。これらの情報は大量に得られることもあるし、少ししか得られないこともあるが、それらを簡潔に(あまり冗長でなく)文章で表現する。

### 3.問題マイクロシナリオ

問題マイクロシナリオというのは、その情報提供者の生活や行動の中に見受けられた問題点を文章によって表現したものである。その問題点は情報提供者自身が問題として指摘するかもしれないし、情報提供者自身は気が付いていないかもしれない。いずれにしても、困っていること、不適切と思われること、苦労していることなど、多様な意味合いにおいて「問題」と考えられることを、一つの問題ごとに一つのシナリオとして文章化したものである。

なお、マイクロシナリオというのは短いシナリオという意味ではなく、問題の微細な構造、つまりマイクロ構造を記述したシナリオという意味である。したがって、シナリオの長さは短い場合には100文字程度のこともあれば、長い場合には400文字程度、もしくはそれ以上になることもある。

また、何ををもって問題とするかは、その調査の目的によって異なる。自動車の運転に関して焦点化した場合、家庭用品の使い方について焦点化した場合、子どもの教育について焦点化した場合、携帯情報機器のあり方について焦点化した場合など、焦点の設定によって問題とすべきことがらは変わってくるし、問題としての書き方も変化する。

### 4.問題マイクロシナリオについての注記

問題マイクロシナリオは、その問題を解決することによって、情報提供者が焦点として設定された側面に関して、より豊かで満足できる生活をするようにすることが最終的な目的としている。人工物の設計によって豊かで満足できる生活が行えるようにするためには、どのような点に注目すべきかを考えながら問題抽出を行ってゆくことが必要である。

### 問題の集約

問題マイクロシナリオをタグによってソートし、共通の問題点を含む複数のマイクロシナリオをたばねる。この操作はソートのキーとなるタグを切り替えることによって複数回反復することが望ましい。

その後、それぞれの問題の束について問題を集約して記述する。

### 解決マイクロシナリオの作成

個々の問題について、考えられる限りの解決案を作成し、それを解決マイクロシナリオ(sMS)として記述する。それぞれの解決マイクロシナリオにはタグとして、その解決案が必要とする技術やそのメリットなどを記述する。

### 解決マイクロシナリオの選定

各々のタグに重要度の重みをつけ、タグとして記述されたポイントとの積和をとり、その解決マイクロシナリオのポイントとする。そのポイントの大きいものが着手すべき解決策ということになる。

## 6 . 単体テスト完了時でのユーザビリティ確認

基本設計段階でユーザビリティについての完全な査閲が完了していればこの段階の修正は極小化することが出来るが、設計書に基づきプログラムが開発された段階で、以下の確認をする。

既存システムからコンバージョンされたデータが既に存在していること、

外部のソフトウェア会社のプログラマーがまだ側にいて簡単に修正の応じられる体制にあることを前提にする。

- ・確認する場合の注意項目は、上記5の紙上検討法と同じであるが、この段階は実際の環境でテスト・確認できるので、平均レスポンスタイムや上限データ量の処理の円滑さなども確認項目とする。
- ・実際の利用者が操作をする時間や考え込んでしまう時間箇所などにも着目し、慣れれば乗り切れる問題なのか？改善すべき課題なのか？を判断する。
- ・特別室に操作テスト環境を作り外部から操作状況をビデオで写しとり観察する方法もあるが、企業内システムでは通常の執務室の一部を活用してのテストで十分である。
- ・eビジネスシステムなど外部の一般の人が操作するような場合は緊張するので、特別な環境を準備することも必要である。

## 7 . ユーザビリティの効果

開発フェーズの中にユーザビリティの実施を位置付け、操作性・利用性の向上をすすめることは次のような効果を生み出す。

- (1) 完成したプロジェクトの操作が使いやすい、利用しやすいものとなっている
- (2) eビジネスについては「使いやすいシステムによる受注増加」が期待できる
- (3) 設計段階からユーザーの意見を合理的に取り入れてあるので、完成したシステムについて

利用者の理解が進んでおり、スムーズな利用開始が可能となる

- (4) 設計段階にユーザビリティの要素が十分に盛り込んであるので、結合テスト、総合テストの段階での修正がほとんどなくなる。したがってこのテスト期間が短縮でき、かつ開発コストの低下が期待できる
- (5) ユーザビリティについてのノウハウが企業に体系的に蓄積されてくると、システムの部品化が進み開発コストの低下が期待できる。またユーザビリティの技術が進歩し、より使いやすい仕組みが検討できるようになる
- (6) 利用部門と開発部門が、システムの利用上の問題についての理解が進み相互理解が深まる。そして新しいビジネスモデルが誕生する可能性が増える

## 8 . 参考 : ユーザビリティチェックリスト

前節では、ユーザビリティの概念とプロセスを学び、その歴史の概観からインターフェース理論、人間中心設計 (UCD)、ISO13407 策定に至る流れを俯瞰した。また、UCD の具体的なメソッドとして「(マイクロ)シナリオ手法」について説明した。

ここでは、「補足 / 付録」としてユーザビリティのチェック項目及びデザイン原則についてまとめる。

前半では、インターフェースを持つあらゆるシステム (ハードウェア及びソフトウェア) に適用できるチェック項目を取り上げる。これらは、研究会における研究過程において提示されたものである。

後半では、ウェブ環境におけるユーザビリティについて、ウェブサイト / アプリケーションの画面デザインの原則についてまとめる。

### (1) ユーザビリティチェック項目

#### a . 操作性

##### 身体適合

- ・ 指や手の大きさ、身体各部の大きさにあっているか
- ・ 指や手の動きやすい範囲に操作部位や可動範囲が収まっているか
- ・ 指や手の自然な動きに逆らった方向への操作をしなければならないようなことがないか

##### 視認性

- ・ 表示部位は必要な情報を表示するに十分な広さを持っているか
- ・ 表示は見つけやすい場所になされているか
- ・ 表示内容が変化した時にユーザーは容易にそれに気がつくことができるか
- ・ 表示文字は小さすぎないか
- ・ 読みやすい書体を使っているか
- ・ 表示文字と背景は適切なコントラストを持っているか
- ・ 表示文字はあまり高密度に表示されてはいないか (適切な字間、行間を用いているか)
- ・ 強調表示 (大きな文字、太字、斜体、下線など) は適切に使用されているか
- ・ 強調表示やリンクは多用されすぎてはいないか

### 可聴性

- ・エラー警告音などは聞き易い大きさ、音の高さ、音色になっているか
- ・通常の警告では一般的な音で、非常事態では多少耳障りであっても気がつきやすい音になっているか
- ・何種類かの音を使っているときは、相互に識別しやすいか、その意味が連想しやすいか
- ・音の大きさはユーザーによって調節可能か

### 疲労軽減

- ・不自然な姿勢を長時間続けることはないか
- ・特定の指だけに負担が集中するようなことはないか
- ・キーが固すぎたり、機器が重すぎたりし、操作していて疲れるようなことはないか

### 携帯性

- ・携帯型の機器の場合、携帯するには重すぎないか
- ・携帯型の機器の場合、携帯するには大きすぎないか
- ・携帯型の機器の場合、不用意に触って誤動作を引き起こす心配はないか
- ・誤動作を防ぐために、操作部のロック（ホールド）ができるか
- ・携帯型の機器の場合、電池の使用時間は十分か
- ・携帯型の機器の場合、電池切れの予告表示機能はついているか

### 収納性

- ・使わないとき、所定の場所に収納しやすく、また取り出しやすいか
- ・小型機器の場合、鞆やハンドバッグなどに収納しやすいか

### 柔軟性

- ・ユーザーが自分の好みに応じた設定をできるか
- ・表示形式を自分のなじみのある形に変更できるか
- ・重要な機能を実行する際に、複数の異なる方法が用意されていて、ユーザーが自分に適した方法を選択できるようになっているか
- ・日本語の入力にはローマ字入力とかな入力が選択できるか

### 効率性

- ・操作の手数は少なく設定されているか
- ・操作の所要時間は短く設定されているか
- ・操作の際の手の動線が自然な流れ（たとえば上から下、左から右）にしたがって設定されているか
- ・熟練したユーザーに対して簡便で効率的な操作法が用意されているか
- ・機器の大きさや利用環境が許すかぎり、情報機器としてはポインティング装置としてマウスを使っており、それ以外の機器の場合には、目的に適合した装置を使っているか
- ・機器操作において（キーボードとマウスのように）手の移動が頻繁に発生することはないか
- ・同時に二つの操作（特定のボタンを押しながら、別の操作をするような）をしなければならないような場合がないか

**エラー対応**

- ・スリップ（操作のミス）を防ぐための配慮がしてあるか
- ・重要な結果を起こすキーは触ってしまいにくい場所に置いてあったりキャップがしてあったりするか
- ・重要な結果を起こすキーは押すのに多少力がいるようになっているか
- ・重要な結果を起こす操作部は単に押すだけでなく、スライドや回転操作になっているか
- ・入力は確実にできるようになっているか、押したか押さないかが確認しにくいことはないか
- ・キーやボタン、マウスなどのクリック感は確実に得られるか
- ・キーやボタン、マウスなどのクリックは軽すぎたり重すぎたりしないか
- ・必要に応じて、キーやボタンをクリックしたときに音によるフィードバックが提供されるか

**b . 認知性****平易さ（知覚関連）**

- ・ゲシュタルト心理学の原理（群化の要因）は効果的に使われているか
- ・類同の要因・類似の機能を同じ色、同じ形、同じ大きさのアイコンやボタンでまとめてあるか
- ・近接の要因・類似の機能のアイコンやボタンを近くに並べ、他のものから遠ざけてあるか
- ・閉合の要因・類似の機能のアイコンやボタンを閉じた矩形などで囲んであるか
- ・目立たせるべき表示（重要な情報など）に図と地の要因を効果的に使っているか
- ・目立たせるべき表示に背景と高いコントラストを使っているか
- ・目立たせるべき表示を他の部分に比べて大きくしているか
- ・複数ウィンドウを表示しているとき、アクティブウィンドウは適切に強調されているか、あるいは最前面に表示されているか
- ・利用可能なボタンはランプがつくとか、強調表示されているか
- ・文字やピクトグラムは読みやすいか
- ・読みやすい大きさや字体になっているか
- ・込み入りすぎていないか
- ・難解な漢字表示、読みのわからない漢字表示を使っていないか
- ・操作の手順は一定の順番に平面的に配置されているか
- ・基本的には左から右に、あるいは上から下に操作するように配置されているか
- ・表示は一見して見やすい形式に配列してあるか
- ・数字は小数点の位置でそろっているか
- ・文字は左詰めにそろえてあるか

**平易さ（認知関連）**

- ・基本的な操作は直感的に分かりやすいか
- ・起動や終了操作は分かりやすいか。
- ・特に終了操作はいつでも直感的にわかりやすくなっているか
- ・分かりにくい用語は使われていないか
- ・ユーザーにとって理解しにくい専門用語は使っていないか

- ・なじみのない記号列や英単語や略称が使われていないか
- ・ユーザーにとって意味のないコード番号や開発用の識別符号が使われていないか
- ・プリントアウトの時、画面と印刷物の間に WYSIWYG の関係が成立しているか
- ・ワープロや Web ブラウザーの場合、完成した状態を確認しながら編集ができるか
- ・ - 操作する対象を名前や座標で間接的に指示するのではなく、マウスなどで直接指示できるか
- ・場合によってはタッチパネルやペンなどによって画面の上で直接操作ができるようになっているか
- ・言葉だけでなく視覚的な表現も併用してあるか
- ・視覚的な表現は内容が直感的に理解しやすいか
- ・視覚的な表現（ピクトグラムやアイコン）だけを利用しているのではなく、言語的なラベルもつけられているか
- ・アイコン表現は必要以上に細かくて判読しにくくなっていないか
- ・ある状態で操作可能な部位と不可能な部位は視覚的に区別してあるか
- ・操作不可能な部位や機能については半輝度にするとか網掛けをするようにしてあるか
- ・操作のための画面部品と表示だけの画面部品は視覚的に区別してあるか
- ・ボタン類については、たとえば立体的に表示してあるか
- ・反対に、操作できない表示箇所をボタンのように立体表示していないか
- ・操作する方向（押す、回転する、スライドするなど）は直感的に分かりやすいか
- ・マウスクリックの位置が入力する数値に対応する場合、微妙な値の指定ができるか
- ・ユーザーが機器のシステムイメージを容易に理解できるようになっているか
- ・複雑な機器であっても、適切なメタファーを与えて容易な理解を促進しているか
- ・メニューの階層構造は適切に機能を分類しているか
- ・メニューの標題からは予想しにくい機能が含まれていないか
- ・メニューがあまりに深く（たとえば4段以上）構成されていることはないか
- ・メニュー選択肢が多いときには、種類ごとに選択肢の間に区切りをいれてあるか
- ・メニューの各項目は平易で内容が予想しやすいか
- ・メニューの項目は、種類、頻度、日付、容量など、任意の基準で整理しなおせるか
- ・数値の表現はわかりやすいか
- ・日常的に理解可能な単位を使っているか
- ・複数の数値表示の小数点位置は縦方向にそろっているか
- ・数値を入力させる場合に、どのような単位によるのかが明示してあるか
- ・例を示すことによって操作の分かりやすさを向上させるようになっているか
- ・図示することによって操作の分かりやすさを向上させるようになっているか

#### 平易さ（記憶関連）

- ・ユーザーが覚えなければならない要素の提示個数はマジカルナンバー（ $7 \pm 2$ ）の範囲内におさまっているか
- ・それを越える場合には適切なチャンキング（情報のグループ化）がなされているか
- ・モードは浅く（たとえば4層以内）設定されているか
- ・モードを識別させるためにモードごとに異なった表示がなされているか

- ・ 7 ± 2 の法則とチャンキング

0 4 2 4 - 8 1 - 1 1 3 6

0 4 2 4 が調布市の市外局番であることを知らない人には 1 0 チャンクの情報、

0 4 2 4 が調布市の市外局番であることを知っている人には 7 チャンクの情報量

- ・ ユーザーの記憶の負担を考慮して、再生型でなく再認型になっているか
- ・ 再生型のコマンドよりはアイコンやメニューなどの再認要素を利用しているか
- ・ ある画面の操作を実行するために、別の画面の情報を利用しなければならないようなことはないか

#### 平易さ (エラー関連)

- ・ 思い違いによるエラーは起きないように配慮されているか
- ・ エラーを犯してしまってもアンドゥー機能 (操作前に戻す) によって元の状態に復帰することができるか
- ・ エラーからの復帰の操作は簡単か
- ・ エラーが起きたとき、その現象の説明 (何が起きたのか) や原因 (機器のどの機能がどうなったからなのか) だけでなく、対処の仕方 (どうすればその状態から脱出できるのか) も説明されているか
- ・ エラーメッセージは簡潔な表現になっているか
- ・ エラーについて詳細な説明が知りたい場合、それを知るための手段が用意されているか

#### 一貫性

- ・ 違った状態やアプリケーションでも、同じ操作は同じ名称、色、形になっているか。
- ・ 同じ機能は複数の画面で同じ位置に表示されているか
- ・ 類似の操作は複数の機能でほとんど同じ手順で実行可能か
- ・ 文字の省略規則や用語の文法規則を一貫させてあるか
- ・ エラーメッセージの与え方や、エラーへの対処の仕方は同じパターンになっているか
- ・ 同じ機能は違った場面でも同じ名称で呼ばれているか。場合によって違う名称で呼ばれていることはないか
- ・ コマンド入力領域は一定の場所、例えば画面の左下の位置、に常に置かれているか。場合によって違う場所に表示されたりしないか
- ・ 入力時のデータの表示形式と、それが表示される時のデータの表示形式が一貫しているか

#### 連想性

- ・ アイコンなどは、ユーザーが慣れ親しんでいる日常的なメタファー (比喩) が利用されているか
- ・ 初めての用語でも、日常生活から容易にその内容が類推できるか

#### 誘導性 (ヘルプ関連)

- ・ ヘルプ機能は適切に提供されているか
- ・ ヘルプの表示される位置は一貫した場所になっているか
- ・ 自動的に起動されるヘルプは表示しないようにもユーザーが設定できるか

#### 誘導性 (ガイド関連)

- ・ 操作手順のガイダンスは行われているか

- ・値を入力する場合、最初にデフォルトないし推奨値が入れてあるか
- ・項目を選択する場合、最初にデフォルトないし推奨する項目が選択されているか
- ・具体例を使って操作が説明されているか

#### 誘導性（ドキュメンテーション関連）

- ・マニュアルには、必要で十分な説明がのっているか
- ・マニュアルには、必要で十分な説明がのっているか
- ・どのような機能が利用可能なのかが容易にわかるか
- ・説明の文章は平易で分かりやすいか
- ・難しい用語は使われていないか
- ・説明は具体的で、実行可能な表現になっているか
- ・適宜、図表をまじえて説明してあるか
- ・図表は込み入っていないで分かりやすいか
- ・本文における部品の説明を図表の中で容易に確認できるか
- ・事例を交えて説明してあるか
- ・操作の手順の説明は実際にそれにしたがって容易に実行できるか
- ・説明が多すぎて、ポイントの把握に時間がかかるようなことはないか
- ・マニュアルは、簡略版、導入編、機能編のように階層的に用意されているか
- ・全体を概観できるようにコンパクトに情報をまとめた資料が用意されているか
- ・初心者、熟練者のように、異なるユーザーを想定して幾つかに区別されているか
- ・下敷き状のコマンド一覧表のようなものが用意されているか
- ・ユーザーが調べたい項目の検索は容易か
- ・索引が用意されているか
- ・機能名称から検索できるか
- ・症状から検索できるか
- ・やりたいことから検索できるか
- ・エラーメッセージからマニュアルの該当箇所が参照できるか
- ・分厚くて携帯に不便ではないか

#### 習熟性

- ・特にチュートリアルソフトを使わなくても自然に習熟できるようなものになっているか
- ・操作の学習を支援するようなチュートリアルソフトが製品に添付されているか
- ・チュートリアルソフトのカリキュラムの進度は早過ぎもせず遅すぎもしないか
- ・チュートリアルソフトは必要と思われる機能範囲をカバーしているか
- ・ユーザーが継続して利用したくなるような配慮がしてあるか

### c . 快適性

#### 主体性

- ・説明の文章の主語はユーザーになっているか
- ・機器が主語になっていて、ユーザーに命令をしているような表現になっていないか
- ・ユーザーが目標を変更したり、間違いに気づいたとき、すぐに中断できるか

- ・システムが故意に情報をユーザーから隠している、というような印象を与えないか
- ・システムは勝手なことをするべきでない

### 寛大性

- ・アンドゥーのような形でエラーからの回復を支援する機能がついているか
- ・エラーを犯したユーザーに厳しい表現、あるいはユーザーに不快感を感じさせるような表現をしていないか

### 美しさ

- ・機器全体のデザインは美しく、統一されたものになっているか
- ・画面レイアウトは整然としているか
- ・画面レイアウトは混雑しすぎているか
- ・機器を利用する環境や利用場面とマッチしたデザインになっているか
- ・オフィスで利用する機器やソフトウェアにはある程度の品格が演出されているか
- ・ゲームのようなエンタテインメント性を持ったソフトには楽しさが演出されているか
- ・色を多用しすぎたために、品のない煩雑なデザインになっていないか
- ・色数は4色、多くても8色に押さえてあるか（ただしグラデーションは除く）
- ・モノクロの画面で見たときにも白黒やグレーのバランスのとれた配色になっているか

### 快適操作

- ・操作に対して即座に応答が返ってくるか
- ・適切なフィードバックがなされているか
- ・操作が行われた瞬間に、入力確認のためのフィードバックが視覚的、聴覚的に与えられるか
- ・処理の完了までの時間はできるだけ2秒以内におこなうようになっているか
- ・処理の完了までの時間がそれ以上なら、タスクバーなどで残り時間が予測可能か
- ・タスクバーを利用する場合、バーの処理済み部分の移動速度は一定になっているか
- ・タスクバーが何回も出現して、全体としていつになったら終了するのかが分からないようなことはないか
- ・無駄な画面（たとえばアニメーションやキャラクター画面）が表示されることで快適な操作が損なわれていることはないか

### 安心感

- ・訳の分からない状態に陥ったりしてユーザーを不安に陥れることはないか
- ・どうしたらよいか分からない状態に陥ってユーザーを困惑させることはないか

### 動機付け支援

- ・ユーザーに使ってみたい気を起こさせるような配慮がなされているか
- ・失敗したときには叱責せず、成功したときにはそれを誉めるようにしているか
- ・楽しく使えるような配慮がなされているか
- ・最初から難しすぎてユーザーに拒絶感を与えてはいないか

### 親近性

- ・メタファーを使うような場合にユーザーの日常生活との連続性に配慮しているか
- ・親しみやすいキャラクターなどを利用しているか
- ・メッセージはアプリケーションの性格に応じた適切な言葉遣いをしているか

#### d . 特別な配慮

< 初心者 / 熟練者 >

##### 初心者一般

- ・ 初心者の立ちあげを促進するように設計されているか
- ・ ユーザーの側がシステムイメージを早く作り上げられるような補助手段が提供されているか
- ・ 初心者が学習しなければならない情報は少なく設定してあるか。すなわち、少し学習するだけでも使えるようになっているか
- ・ 電話相談のシステムが用意されているか
- ・ 電話相談はユーザーが利用したい曜日や時間帯に設定されているか
- ・ 電話相談の回線がいつも塞がっているようなことはないか

##### ハイテク弱者

- ・ ハイテク機器が苦手なユーザーにも、使ってみたいという気を起こさせる配慮があるか
- ・ そうしたユーザーが使おうとしたとき、とまどいや困惑を覚えることはないか
- ・ ユーザーに不安を与えて、もう懲りた、と思わせてしまうようなことはないか
- ・ ユーザーに機器を壊してしまうのではないか、との不安感を抱かせてしまうようなことはないか

##### 低位頻度利用ユーザー

- ・ たまにしか利用しないユーザーでも、操作を忘れずにいられるか
- ・ 操作を忘れてしまっても、直感的にすぐ操作できるか

##### 熟練者一般

- ・ 熟練者にふさわしい機能が用意されているか
- ・ コントロールキーなどを使ったショートカットが用意されているか
- ・ 自分なりの操作環境に設定しなおすためのカスタマイズ機能が用意されているか
- ・ ショートカットの割付けなどは自分で設定することも可能か

##### 長期利用ユーザー

- ・ 長い間利用していれば、自然に効果的な使い方を身につけることができるか

##### 高頻度利用ユーザー

- ・ しょっちゅう利用していると、わずらわしいとか面倒だと思えてくるような所はないか

##### 専任オペレータ

- ・ 機器を専門に扱うユーザーに対して専門のトレーニングの機会が用意されているか

#### e . 特別な配慮

< 特別な配慮を必要とするユーザー >

##### 視覚障害

- ・ 視覚障害の人にも利用できるか
- ・ 聴覚による情報提示を、視覚的な表示と合わせて行っているか
- ・ 点字や触覚を利用した手がかり、音声の利用など、代替方法が用意されているか
- ・ 複数のボタンがある場合、基準となる位置のボタンに突起をつけるようにして基準の枠組みを

提供しているか

- ・危険な場所に指をはさんでしまうような可能性はないか

#### 聴覚障害

- ・聴覚障害の人にも利用できるか
- ・視覚による情報提示を、聴覚的な表示と合わせて行っているか
- ・視覚表示の急激な変化によって、聴覚障害を持った人にショックを与えないよう配慮しているか

#### 身体障害

- ・長い文字入力を必要としないようになっているか
- ・手指に障害のある人にも無理なく使えるか
- ・運動障害の場合、キーガードのような装置により、目的のキーを確実に押せるか
- ・同時入力操作を順次入力操作に切り換えることができるか
- ・キーのオートリピートをオフにすることができるか
- ・マウス操作をカーソルキーでもできるように冗長に設計してあるか
- ・左右半身麻痺の人にも無理なく使えるか
- ・利き手以外の手をつかっても操作に困難を覚えることはないか
- ・下半身運動障害の人にも無理なく使えるか
- ・車椅子から操作することが容易になっているか

#### 幼少児

- ・幼児にも使えるべき機器の場合、無理なく使えるか
- ・表示の表現は幼児にも理解できるか、難しい漢字表示を使っていないか
- ・児童にも使えるべき機器の場合、無理なく使えるか
- ・操作部位は児童の手の大きさに適合しているか
- ・表示の表現は児童にも理解できるか、難しい漢字表示はないか

#### シルバー世代

- ・シルバー世代にも無理なく使えるか
- ・画面やマニュアルの文字表示は小さすぎないか
- ・画面表示の拡大機能があるか
- ・特定の色に対する感受性の低下に対する配慮はしてあるか
- ・記憶力の低下への対処として、ユーザー側の記憶の負担を軽減してあるか

#### 左利き

- ・左利き右利きのどちらのユーザーにも無理なく使えるか
- ・操作部位は左右対称になっているか
- ・左利きユーザーのためのカスタマイズ機能がついているか

#### 色盲

- ・色盲のユーザーにも無理なく使えるか
- ・赤と緑、青と黄の組み合わせを識別するための表示に使っていないか（補色関係にある反対色よりは少しずれた色の組み合わせの方が良い）
- ・機能の識別に色を使っている場合、色だけでなくラベルも併用しているか

### 異文化圏のユーザー

- ・特定の文化圏での行動パターンに適合しているか
- ・緊急時にドアやレバーを引くのか押すのかが分かりやすいか
- ・表示に（少なくとも）英語を利用あるいは併用しているか
- ・データの表示形式は、利用文化圏に適合しているか
- ・日付（年月日）の表示は利用文化圏の形式になっているか
- ・特定の文化圏での感性に適合しているか
- ・ユーザーの属する文化圏で不愉快な感情を引き起こさないような色を使っているか

## （２）ウェブ上のユーザビリティに関する原則

### a . アイデンティティ

システムの存在意義を表現するための重要な要素が「アイデンティティ」である。システムにアクセスしたユーザーが、「何を目的としているのか」「何ができるのか」といったことを把握できなければそれを使い始めることができない。アイデンティティをきちんと表現することによって、ユーザーにとってのシステムを利用価値や統一感のあるイメージを訴求でき、システム全体を通じた信頼性の向上にも役立てることができる。

#### システムの目的が明確になっていること

システムにアクセスしたユーザーは、そこでできることを速やかに把握し、自分の求める情報や機能がそこにあるかどうかを判断しようとする。その期待に応えるためには、そのシステムの目的を端的に、そして明確に表現する必要がある。一見してもシステムの目的を表現するヒントが見当たらなかつたり、利用を進めてもその用途や目的を理解できなければ、ユーザーは最終的に自分の目的を達成できず、途中でそのシステムの利用を放棄してしまう恐れがある。

運営者名の明示、システムの名称や ID、タスクの開始点となる機能がはっきりと把握できるようなメニューなどを配置し、ユーザーがそれぞれの役割をきちんと理解できるように表現する必要がある。

#### ブランドのルック&フィールが適切に表現されていること

システムの提供者や運営者のコーポレートアイデンティティ、製品・サービスのブランドを表す名称などが適切に表現されていると、ユーザーはそのシステムの機能や運営者が持つコンセプトを端的に把握することができる。また、システム全体を通じて、それらの要素が統一感を持って訴求することで、ユーザーの中で築かれるブランドイメージやコンセプトをより深く浸透させ、システムに対するロイヤルティを継続的に、また効果的に向上できるようになる。

たとえばウェブ上では、既存メディアで用いてきたブランドロゴを特定のサイトで用いたり、逆にそのシステム独自の VI( ビジュアルアイデンティティ )を強調して表現するなどといった工夫を施すことができる。また、システム全体で基調色を用いた共通デザインなどを設けることで、ユーザーがそのウェブサイトを利用するたび、確固としたブランドイメージを定着させていくことができる。

### コンタクト手段が十分に提供されていること

システムの提供者は、ユーザーが運営者に対して「連絡を取りたい」という積極的な気持ちを重視しなければならない。なぜなら、その積極性はシステムの価値向上や運営者の最終的な利益に直接結びつくだけでなく、運営者にとって、ユーザーのシステム利用に対する満足度を向上させるための材料を入手するための絶好の機会にもなり得るからである。その機会を最大限効果的に得るためには、ユーザーが速やかに利用できるようなコンタクト手段を十分に提供する必要がある。

様々なインタラクティブメディアの中でも、ユーザーとの直接的なコンタクトを送受信できることはウェブを用いることの大きな特徴であり、そこでは独自のコミュニケーションが生まれる。一方、ユーザーにとっては、特定のウェブサイトがその運営者に接する唯一の方法であることも多い。ユーザーが連絡を取りたいと思ったときに、ストレスなくコンタクトできるよう、問い合わせ先を分かりやすく明示したり、電話番号・メール・フォームなど複数の問い合わせ手段を提供しておく必要がある。

### 信頼性が高いこと

システムに対する信頼性を高め、ユーザーが安心して利用できる状態を保つためには、そのシステムやサービスの管理責任者や運用者の所在を明らかにし、具体的なコミュニケーションの相手として認識できるようにしておかなければならない。多くのユーザーは管理責任者の存在を把握するだけでなく、その存在が信頼に値するかどうかを判断するための材料を求めている。運営者の背景に関する十分な情報や運営方針などをきちんと明示し、常にユーザーと信頼関係を結ぼうとする姿勢を示していくことが重要なのである。

特に、ユーザーに対して個人情報の提示を求める場合は、システムの信頼性をより強調することに細心の注意を払う必要がある。たとえば、プライバシーポリシーをはじめ、ユーザーとの間で送受信される情報の取り扱い、利用上の注意事項、運営者側の免責事項などを、すぐに確認できるようにしておかなければならない。

## b. インフォメーションアーキテクチャー

システム内の情報を整理して、ユーザーが理解しやすく構造化することが「インフォメーションアーキテクチャー」である。膨大な情報があっても、その中から必要なものをすぐに見つけ出せなければ意味がない。様々な情報や機能を、ユーザーにとってのわかりやすさを配慮しながら整理し、論理性を確保した適切な状態で表現する配慮が求められる。

### 分類と階層が適切であること

人が身の回りの物を探している場合、どこにどういった分類でそれが格納されているかが把握できていれば、短時間で問題解決に至ることができる。これはあらゆるシステムにおいても同様である。ユーザーの情報検索効率とタスク達成度は、そのシステム内のコンテンツや機能がどのように分類・階層化されて提供されているかによって大きく左右される。ユーザーにとって分類の基準が分かりにくかったり、階層が深すぎたりすると、目的の情報に辿りつけなかったり、数多く提示される条件分岐の途中で誤った選択を行う確率も高くなってしまうことになる。

ユーザーは、たとえばそのシステム上で提供される主要なメニューを見て、全体の構造や機能の概要やそこに蓄積されている情報群の分類基準を把握しようとする。それらが論理的に、なおかつ整理された状態で表現されていれば、速やかに必要な情報に辿りつくことができる。一般的に、重要な情報や機能を提供するセクションは、情報量も膨大になりがちであり、ユーザーがなかなかタスクを完了できない場合が多い。そのような点も十分に配慮し、常にユーザーの目的に合った、バランスの良い情報構造を配慮する必要がある。

### ラベルが分かりやすいこと

人が身の回りの物を探している場合、どこにこういった分類でそれが格納されているかを把握しようとしても、その分類内容そのものがきちんと明示されていなければ、短時間で問題解決に至ることができない。システムにおいて、コンテンツ、カテゴリー、機能やボタンなどの名称として用いられている文言を「ラベル」、文言を付与する行為を「ラベリング」と呼ぶ。情報構造を分かりやすくするためには、ユーザーが理解しやすい文言を用いたラベルを付与することが重要である。

システムで用いられる主要なメニュー項目や個々の機能の名称などに、特定のユーザーにしか理解できない用語や造語などが使われている場合、ユーザーはその機能を使用する前に、ラベルの意味をよく考えなければならない。ユーザーは、そのような不要なストレスを与えるシステムに対して、途中で利用を中止してしまう恐れがある。各ラベルにはユーザーにとって最も分かりやすい文言を採用することが重要なのである。ただし、システムの用途によってはユーザー層が限定され、その背景が異なる場合もあり、「最も分かりやすい」ラベルの基準が変わってくるため、対象となるシステムにおけるラベリングの方針を検討する配慮が必要である。

### レイアウトが効果的であること

平面や空間に複数の要素が存在する場合、人は個々の要素の大きさ・形・色・位置などといった特徴、各要素間の相対的な距離、複数の特徴から生成されるパターンなどを手掛かりに、その事象の意味や関係性を推測する。システムにおいても同様に、ユーザーは自分が目にしている画面のレイアウトから、そこで提示されている要素や情報の重要度、その関連性などを把握しようとすることになる。特に画面上の文章はきわめて読みづらいものであり、内容の把握に時間が掛かる。文章を細かく段落分けし、見出しや箇条書きを用いて視覚的にパターン化したり、文章を最も重要な結論から先に書き始める「逆ピラミッド型」の構成を用いるといった工夫をし、ユーザーができるだけ短い時間でその要点を把握できるようにしておく必要がある。

たとえば、ウェブ上のシステムの場合、ユーザーは左から右、上から下へ視線を移動するため、ページの左上端などに最も重要な要素を配置すると効果的である。逆にページの最下部など、スクロールしなければ目にすることのない位置に配置されている情報については、重要でないどころか、その存在を認識すらされない恐れがある。加えて、ブラウザの種類やウィンドウの面積などといった閲覧環境が変化するようなことが前提になっている場合は、特定の環境に依存せず正しく情報が提示されるよう、柔軟性を持ったデザインが求められる。

### ドキュメンテーションが十分であること

ユーザーがシステムを利用する際には、何も悩むことなく自分の目的を達成できることができれば理想的である。しかし、多くのシステムの用途が多様化し、その機能性が複雑化している現実において、ユーザーが何も疑問を持たずに作業を継続できることは困難に等しいとも言える。そのため、ユーザーが必要とする際にすぐに参照できるような「ヘルプ」「FAQ」などといったドキュメンテーションが果たす役割は大きい。適切なドキュメンテーションを用意することは、ユーザーの利便性を重視しようとする運営者の姿勢を直接的に示すだけでなく、問い合わせ対応に掛かる費用を節約する効果もある。

たとえば、システム全体あるいは特定の機能に関して、利用方法や過去に受けた質問とその答え、適切な関連情報などが用意されていると、利用中に操作に迷ったユーザーや、より詳しく知りたいと考えるユーザーにとって大きな手助けとなる。特に、そのシステムを初めて利用するユーザーは、まずその情報にアクセスする可能性が高い。ドキュメンテーションの利用しやすさが、そのシステムを繰り返し利用するための重要な動機づけになると考えることもできる。また、ウェブにおいては、サイト全体の情報構造を視覚化したサイトマップや、「五十音順」「ジャンル別」などユーザーにとって意味のあるいくつかの切り口を索引として用意しておくことで、ユーザーの情報検索効率を向上させることができる。

## c. インタラクション

ユーザーが目的を達成するためにシステムとの間に持つ相互作用が「インタラクション」である。各種コントロールの振る舞いや画面遷移がユーザーのタスクに合っていないければ、効率的な操作ができない。ユーザーが自分の目的を速やかに達成できるためには、そのシステムを間違えることなく、また、最も効果的に利用できるようにデザインする配慮が必要である。

### 直接操作感と適切なフィードバックが提供されること

ユーザーが自分の思考過程に沿って自然に操作を行い、思いどおりにシステムを利用するためには、提示されているリンク、ボタン、テキストなどといった要素や機能を「直接操作しているように感じさせる」ことが大切である。直接操作の感覚は、ユーザーが行った操作に対するフィードバックが、即座に期待どおりの形で帰されることで得られるものである。

ウェブ上では、ユーザーとシステムが相互にコミュニケーションをとって機能することが前提となっている。特にそのようなシステムにおいては、ユーザーが直接操作の感覚を持つことができなければ、ひとつひとつの操作に際して絶え間ない判断を強いられ、結果として操作の継続が困難になる恐れがある。ユーザーの操作に対するフィードバックを即座に分かりやすく提供することで、ユーザーにとって違和感のないシステムであることを印象づけることができるようになる。

### 可認性と知覚的安定性が高いこと

ユーザーが自分の思考過程に沿って自然に操作を行い、思いどおりにシステムを利用するためには、提示されているリンク、ボタン、テキストなどといった要素や機能を「直接操作しているように感じさせる」ことが大切である。直接操作の感覚は、ユーザーが行った操作に対するフィードバックが、即座に期待どおりの形で帰されることで得られるものである。

ウェブ上では、ユーザーとシステムが相互にコミュニケーションをとって機能することが前提となっている。特にそのようなシステムにおいては、ユーザーが直接操作の感覚を持つことができなければ、ひとつひとつの操作に際して絶え間ない判断を強いられ、結果として操作の継続が困難になる恐れがある。ユーザーの操作に対するフィードバックを即座に分かりやすく提供することで、ユーザーにとって違和感のないシステムであることを印象づけることができるようになる。

#### 寛容性とエラー回避性が高いこと

ユーザーは失敗を恐れるものである。また、自分が行った行為が無駄になってしまうと、それまでの積極的な気分を一瞬にして失ってしまう恐れがある。ユーザーがそのシステムを初めて利用する場合、あるいは利用したことがない機能などに直面した場合、さらにウェブの利用経験が少ないユーザーが利用している場合には、その傾向が顕著である。

間違った操作でタスクの続行が不可能になってしまったり、苦勞して入力した内容が無駄になってしまわないように、できる限りエラーを未然に防ぐような工夫を施す姿勢が求められる。また、万が一エラーが発生した場合にもその対処方法を丁寧に示すことで、ユーザーは躊躇することなくそのシステム内を移動したり、使ったことのない機能を試してみることができるようになる。これらの配慮を施すことは、ユーザーに対してシステムの機能性に関する学習を促進し、最終的にそのシステムの利用価値を訴求するうえでも重要なことである。

#### タスクに合った操作フローが提供されていること

システムを利用するユーザーの多くは、何らかの目的を持っている。あらかじめユーザーのタスクを適切に想定し、自然なナビゲーションによって目的が達成できるよう、タスクの流れに沿ったステップとメニューが提供されていることは重要である。

たとえば、オンラインサービスで書籍購入の手続きにおいて、すべての情報入力を終えた最後のページで、初めて「このサービスは会員限定でありこのまま購入できない」ことに気づいた場合、ユーザーはそのページ遷移や情報提供の流れが理不尽であると感じてしまうだろう。さらに、多くの商品の購入プロセスなどでは、作業を途中からやり直したり、情報を保持したまま直前のステップに戻ることができないことも多いため、結局ユーザーのそれまでの行為が無駄なものになってしまうことも多い。このような混乱状態を避けるためには、ユーザーのタスクに沿った操作フローを提供するとともに、ユーザーが自由にサイト内をナビゲーションできる機能を用意しておく必要がある。また、ユーザーの利用を適切に支援するためには、システム全体の情報構造や各ページ内の要素は、静的な論理性だけでなく一連のタスクに沿った動的な利便性を考慮してデザインされる必要がある。

#### d . アクセシビリティ

システムへのアクセス可能な度合いを高く保つことが「アクセシビリティ」である。メディアの特性から生じる利用上の制約を理解し、操作の互換性と情報に対するアクセス機会の平等を最大化することを目指すなければならない。アクセシビリティの視点から、一人でも多くのユーザーがそのシステムを適切に利用できるように配慮することで、結果としてそのシステムの公共性が確保され、さらにはウェブというメディアの存在価値を最大限に訴求できるようになる。

### 閲覧環境に適していること

システムにおける「アクセシビリティ」確保の第一歩は、まずそのシステムにアクセスできるようにすることである。ただし現実には、ユーザーが置かれている利用環境は様々であり、その第一歩を踏み出せないことも多い。特にオンラインサービスにおいては、通信回線の状態やハードウェアのスペックなどといったインフラ環境をはじめ、OS やブラウザの設定、プラグインの有無などによって制約が発生する場面がある。また、障害者や高齢者が音声読み上げソフトなどの特殊な支援技術を利用している場面を考慮する必要もある。いかなる場合でも適切に情報や機能を提供するためには、閲覧環境に影響を与える制約の特性を理解し、表現方法を最適化しておかなければならない。

たとえば、ウェブアプリケーションにおいては、表示スタイルは文書構造と適切に分離したり、特定の入力デバイスに依存せず少なくともキーボードを用いたすべての操作を保証し、また、フレームやレイアウトテーブルの使用についてはそのデメリットを十分踏まえた上で検討する必要がある。

### 情報の存在を認識できること

たとえシステムにアクセスできても、提供された要素そのものの存在を認識できなければ、ユーザーは何も情報を得ることができない。文字、画像、音声、動画など様々な表現手段を用いた情報ひとつひとつについて、存在をはっきりと認識できるようにするだけでなく、ユーザーが視覚、聴覚、動作面などに制約を受けている場合でも、同等の情報を入手できるようにしておく配慮が必要である。

たとえば、視覚に障害があった場合、画像情報を認識することはできない恐れがある。また、聴覚に障害があった場合、音声が生再生されていることを認識できない可能性が高い。まず根本的な問題解決策として、テキスト以外の形式で表現されている情報については代替情報を提供したり、同等の情報や機能を利用できるよう代替手段を提供する必要がある。

### 内容を理解できること

たとえ情報の存在を認識できても、提供された情報の意味が通じなければ、ユーザーは情報の価値判断を行うことができない。システム上で各要素の意味を正しく伝えるためには、情報の構造を論理的に明示し、正しい言葉で情報の重要度を表現することができれば理想的である。また、特定の制約を受けているユーザーには違う意味に取ることができるような要素が存在する状況は避けなければならない。

また、専門用語や読みが難しい言葉やなじみのない外国語の多用を避けることも重要であるが、その文言の利用を避けられない場合には、積極的に理解を促すような補足情報の表現を工夫し、ウェブアプリケーションにおいては文書構造を明確にマークアップするなどといった配慮も必要である。

### 思うように利用できること

たとえ特定の情報のみを認識し理解することができても、その後、そのシステムの利用を阻害

されることなく閲覧行為を継続できなければ、ユーザーがウェブサイトを有効に利用している状態とは言えない。閲覧行為を阻害する要因は排除し、万が一、問題が発生した際にもユーザーが自分で状況を理解し、問題解決できるようにしておく必要がある。運営者側の都合でユーザーの行動を制限することは避けるべきである。

たとえば、動画や音声を再生する場合には、ユーザーが再生・停止をはじめとした制御を行うことができるコントロールを併せて提供する必要がある。また、音声読み上げソフトのユーザー向けには、何度も繰り返し読み上げられるリンク群をスキップするためのリンクを提供するなどといった配慮も望まれる。

## e . パースエージョン

システムが演出するコミュニケーション形態によって、ユーザーの考え方や行動を導くことが「パースエージョン」である。ユーザーの利益を優先し、作業の代行や効果的なガイドによって実現する。単に、ユーザーにとって必要な機能を提供するだけでなく、運営者がより積極的にユーザーにとって効果を提供するような仕組みをデザインできれば理想的である。

### 手続きが簡便化されること

ユーザーにとっての利用価値を高めるためには、本来複雑な手続きが必要である要求をシンプルな操作で実現し、作業コストに対する利益率を高めなければならない。これはそもそも「なぜその製品やサービスを作るのか」という基本的なビジネスモデルに関わることであるが、それだけでなく、同じ結果を得るのであればユーザーはより楽なほうを選ぶという一般的な現象に着目した考え方でもある。ユーザーは、自分の利益に見合ったコストしか負担しようとししない。

ユーザーの手続きが簡便化されるということは、システム側がそれを代行するということである。たとえば、従来、電車の路線や乗り換え方法を知るには時刻表を苦労して調べなければならなかったが、それがオンラインのサービスで簡便化されるようになった。ユーザーは駅名や時刻などの必要な情報を入力するだけで、あとの複雑な作業はシステムが短時間で自動的に行う。作業コストは最小であり、そこで得る結果は要求に対して最大であるため、多くの人々が高い利用価値を感じるものとなる。もし入力項目が何十項目もあったり、結果が表示されるまでに何時間もかかったりすれば、利用価値はほとんどなくなってしまう。

### ガイドとレコメンデーションが提供されること

一般的にユーザーは、完全に自由な意思決定を求められるよりも、あらかじめ用意された選択肢の中から適当なものを選ぶ方を好む。ユーザーにとって有益な結果を想定し、適切なガイダンスとしてタスクの進行を促すことが重要となる。これは特に、ユーザーとシステム双方の目的が明確で、必要な作業がパターンとして抽出できる場合に有効であり、ウィザード式の操作フローによって実現される。

また、ユーザーの趣向や意思決定プロセスを統計的にパターン化できる場合には、レコメンデーションとして、パーソナライズされた選択肢を提供することも有効である。たとえば、オンラインショッピングのウェブサイトでは、類似した行動履歴を持つ他のユーザーが選んだ商品や、かつて購入した商品と関係の深い商品などを適宜レコメンデーションすることは、ユーザーの満足

度を向上させることになり、有益な経験を提供することになる。

しかし、ガイドやレコメンデーションを提供する場合には、倫理的な側面に配慮しなければならない。本来開示可能な選択肢が意図的に制限され、タスクの進行が特定の方向に誘導されていることを、ユーザーが正しく認識できるようにしなければならない。

### 原因と結果のシミュレーションが可能であること

ある要求に対する結果をユーザー自身が予測できない場合、それらをシミュレーションできると有益である。シミュレーション機能はコンピュータの用途として古典的なものであり、特定のアルゴリズムに従って膨大な計算を行う能力は、人よりもずっと高い。そのため、現実には試してその経緯を把握することが困難な場面であっても、条件の指定とともに擬似的に結果だけを抽出することで、ユーザーにとっての利用価値を生むことができる。

たとえば、製品やサービスに関するオンラインでの見積りシステムでは、複数の商品やオプション選択によって価格の合計がどのように変化するかを試算して比較することができる。あるいは、商品の外観や利用状況が条件によってどのように変化するかを画面上で確認することができる。このようなシミュレーション機能は、ユーザーの意思決定にとって重要な手掛かりを与えると同時に、ユーザーの体験を通して能動的な行動を促すことにつながる。

### 親しみやすさと礼儀正しさが感じられること

ユーザーはシステムの利用価値を、提供者のビジネスモデルとは関係なく、自分にとって有益なコミュニケーションの相手かどうかで判断する。そのため、まずユーザーの利益を優先する姿勢がデザインによって示されていなければならない。これは現実世界における接客行為と同じように、親しみやすさと礼儀正しさという態度によって表現される。

つまり、デザイン上の不具合から生じるエラーをユーザーのせいにしたり、技術上の都合から生じる恣意的な規則をユーザーに強制してはならない。問題が起きた場合には、システムの能力不足としてユーザーの解決行動を促す必要がある。たとえば、半角数字の入力が必要な箇所に全角数字が入力された状態でサブミットボタンが押された場合、それを「エラー」としてユーザー側のミスであるように表現するべきではない。システムにとってはエラーであっても、ユーザーは正しい行動を取ろうとしただけなのである。この場合、まず入力文字の全角と半角を自動的に変換する機能をシステム側に持たせ、次に不用意にサブミットボタンが押されないように無効値が入力されている状態ではディスエーブルにするなどし、それでも問題があった場合には、エラーメッセージではなく、再入力という追加作業をユーザーにお願いするような表現を取らなければならない。

親しみやすさや礼儀正しさとは、言葉づかいのことではなく、その真摯で誠実な振る舞いのことである。これを実現するためには、十分な情報、柔軟な対応、そして美しいデザインが必要となる。

