プロジェクトマネジメント（コミュニケーション文書編）

～文書を通したコミュニケーション～

2010年3月　PMFactory

インデックス

どうしたら伝わるのか

“確実にキャッチされる球を”

「キャッチボールで大事なのは投げることではなく、受けてもらうこと。

話すことも書くこともそれと一緒で、情報量を増やしても、伝わらなければ意味がない。

いくら豪速球を投げてもダメなんです。

「伝える」ではなく「伝わる」に意識を置かねばならない。」

（山川静夫　エッセイスト・元NHKアナウンサー）

みんなの生の声

☆誰のために書かれたものか分からないドキュメントが多い。作成者以外が見ても分かる内容・レベルになっていない。

☆納期が間に合わなくなったらテスト作業やドキュメントを省略してしまう。

☆内容が更新されていない。

☆システムの全体を表した資料がない。

☆分かりにくいドキュメントばかりしかない。

☆機能を実現するために必要な情報が設計書に記述されていない。

☆客先要件の実現方法が明確に書かれていない。

☆開発内容に対するコンセプトや背景や経緯についての資料がない。

☆システムの全体を表した資料がない。

☆品質は上流工程（要件定義書・設計書）で決定する。テストだけでは品質は改善しない。

☆客先要件や仕様内容の記述が乏しくロジック記述に偏っている。運用テストに使えない。

☆障害解析に使用できるレベルの内容になっていない。

☆障害報告書で技術的側面から図表等を使用した報告ができていない。

☆モジュール間の連携部分の説明が貧弱。

☆既存ソフトの流用の可否を判断できる資料になっていない。

☆整理・管理されていないため調査・検索ができない。

以上を読んでいると目の前が真っ暗になってきます。

これでまともに動くものができるのでしょうか。

なんでそうなるの？

☆前任者がドキュメントを更新していなかったから、自分もできなかった。

☆協力会社に設計以降の工程を丸投げする方が安く・納期短縮もできたから。

☆バージョンアップ開発時にはソースベースで調査・改修でも何とかなった。

☆発注元からドキュメントを要求されることもなくドキュメント作成工数が削減できた。

☆発注元が協力会社に丸投げする状況が続き、発注元の担当者が自ら設計する技術がなくなってきた。

☆時間に余裕がなく、とりあえず自分が分かるレベルの内容記述だけになった。客先・SEや他の開発者・協力会社が理解できる内容にはなっていない。

☆時間も人もいなくて、限定した内容だけの記述しか作成できなかった。更新が必要な資料にも手をつけられなかった。

開発文書は先につくるものじゃないの？

大切なのはプログラムですか？　ドキュメントは一応作っておけばいいものですか？

本来、要件定義書や設計書は何のためにあるのでしょうか。

開発ドキュメントは、開発者が正しく設計・開発できるように客先要求を開発の言葉に変換したものです。何をどういう風に作れば良いかを可視化したものであったはずです。

決してプログラムを作りながらとか作った後に作るものではないはずです。

開発ドキュメントは目標にたどりつくための地図のようなものです。

先に地図がなければ目的地には到達できないでしょう。どこの世界に目的地らしい所についた後で、後付け的に地図をつくる人がいるでしょうか。

そのような人やチームがあれば道に迷うか遭難するかのどちらかです。

開発文書はコニュニケーションか？

我々開発者が取り扱う文書は主に開発文書および業務文書の二種類がありますが、いずれにしても文書は会話と同様にコミュニケーションの重要な手段のひとつです。

文書（書き言葉）と会話（話し言葉）の特徴は下記の通りです。

●伝達の特徴

文書； 文字（含む図・表）のみの伝達でもっぱら意味の伝達の役割に集中する。抽象度が高い。

会話； ボディランゲージ・態度・強弱等を使用し感情の伝達に優れている。抽象度は低い。

●記録性、正確性、複雑性、伝達量

文書； 記録性に優れている（ノンボラタイル）。 正確。 複雑な情報の伝達可能。 伝達量は大きい。

会話； 記録性に劣る（ボラタイル）。 不正確。 複雑な情報の伝達不可。 伝達量は小さい。

●時間・空間（場所）の同一性、伝達対象者数

文書； 時間・空間の同一性の制限なし。 ほぼ無数の相手に伝達可能。

会話； 時間・空間の同一性の制限あり。 伝達対象者は制限される。

上記、それぞれの特徴から明白なように、プロジェクトの人間系の交流には会話を用い、開発情報の伝達には文書を用いる理由を再確認して下さい。

会話もおろそか、文書も貧弱な開発組織は組織として成立しません。

開発業務におけるさまざまな文書(ドキュメント)

1. 開発文書(技術文書、テクニカル・ドキュメント)

・要件定義書

・設計書(概要設計書・詳細設計書・業務運用フロー・データフロー・システム論理構成図・プロセスフロー・ソフト構造図・修正影響度表・インターフェース仕様書・構成管理手順書 等)

・企画提案書、見積書、開発計画書

・開発管理表(予算管理表・進捗管理表・労務管理表・機材管理表・成果物管理表・障害管理表等)

・ソースコード(コンピュータ言語記述書)

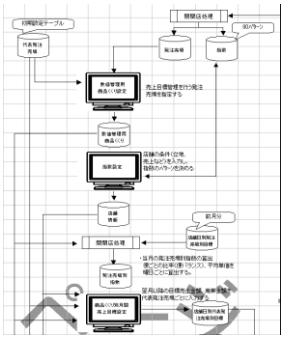
・技術メモ等

2. 業務文書(ビジネス文書、ビジネス・ドキュメント)

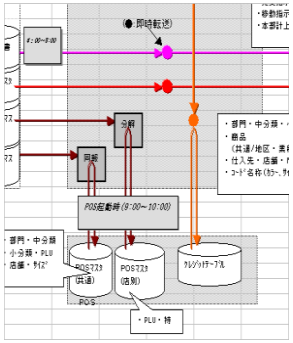
・報告書、連絡書、議事録、ビジネスメモ等

基本設計資料(例)

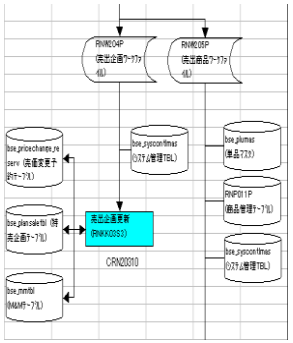
業務運用フロー



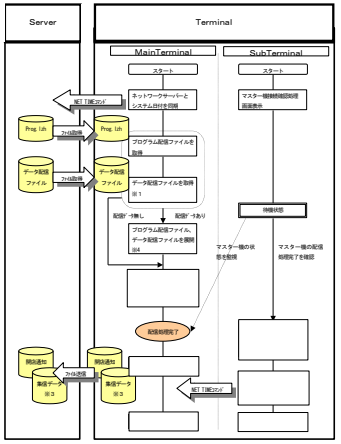
データフロー



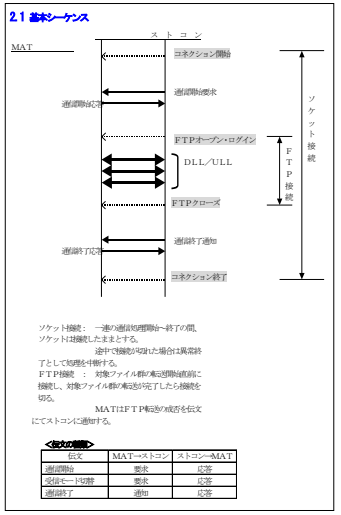
プロセスフロー



機能仕様書

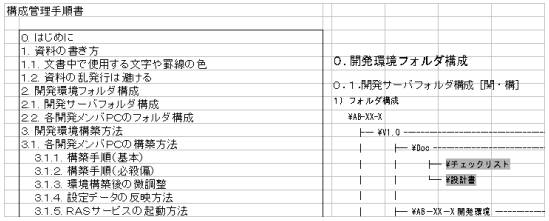


インターフェース仕様書

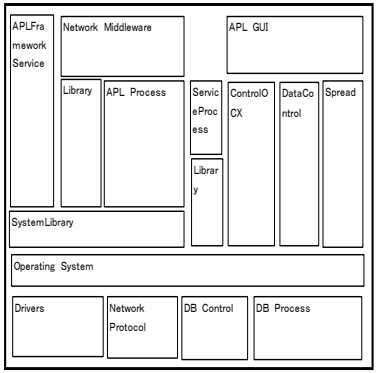


修正影響度表

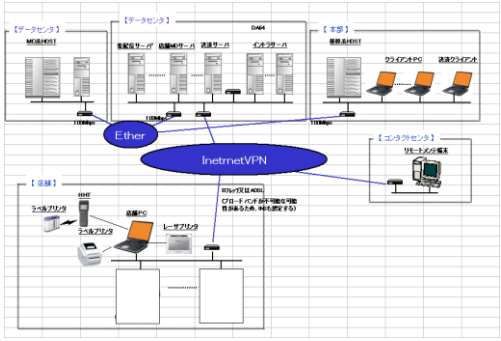
構成管理手順書



ソフトウエア構造図



システム論理構成図



「設計図がない」状況からの脱却

『 いきなりコーディングを始めていませんか。設計図も作らずに 』

多くのエンジニアリング分野では、モノを作り始める前に、まずは設計図があります。

実は現在、ソフトウエア開発において「設計図」らしきものはないに等しいと思われます。

あるのは100万行を超える膨大なソースコードの束だけでしょう。

ソフトウエアはコンピュータ誕生の当初から、コンピュータに対する命令が文字で表現されてきました。いわゆる機械言語と呼ばれるもので、以来、機械語、アセンブリ言語、C言語、Javaと高度化しましたが、常に文字で書かれた「文章」でした。

難解なソースコードを読めるのが一人前のソフトウエア技術者であるとの認識が業界の常識でした。

出典;「設計図がない　脱プログラミング至上主義」(日経エレクトロニクス 2006.9.11　進藤智則)

「設計図がない」状況からの脱却

しかし、時代は変わりました。組込みソフトウエアの規模は2000年ごろを境に100万行を超えました。 携帯電話に至っては1000万行に達する勢いです。既に一人の技術者が全貌を把握できる規模ではなくなっています。

今必要なのは、開発にかかわる技術者すべてが全体や細部の構造を把握できるようにするための、ソフトウエアの設計図です。

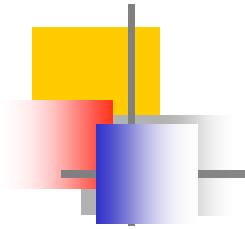
文章だけではなく図で表すことで、技術者の理解度は段違いに高くなります。

今後、ソフトウエアの開発は、プログラミングではなく、設計図（モデル）を作ることを意味するようになります。

パソコンの前で、徹夜でプログラミングのためにキーボードをたたき続けることだけが、ソフトウエア技術者の仕事ではないのです。

こうした設計図中心のソフトウエア開発を実現するための手法が「モデル・ベース開発」です。

出典;「設計図がない　脱プログラミング至上主義」（日経ｴﾚｸﾄﾛﾆｸｽ 2006.9.11　進藤智則）



「設計図がない」状況からの脱却

事実、1999年当時のPOSレジスターにおいてもすでに100万行を越えていました。もうソースコードからソフトウエアの構造を読み解くことはほとんど不可能になってしまいました。

ソフトウエアが大規模化・複雑化していく状況において設計図も構造図もない状態では多数の技術者・評価担当者・他のプロジェクトの開発者等にソフトウエアの構造・機能を説明し理解してもらう事は全く不可能になってしまいました。

コスト圧力が一層の厳しさを増し、オフショア開発が急激に拡大する中でプログラミング作業はすべて中国を筆頭にした国々に渡り、日本に残れるソフトウエア開発組織における価値のある仕事の一つは要件定義書および設計図（モデル）の作成なのではないでしょうか。

ビジネスピンチ？それともチャンス？

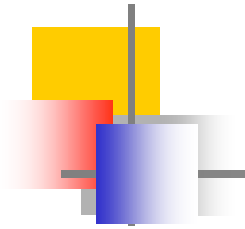
設計図・構造図を書けない・書かないためどれだけの損失が発生しているのか試算してみて下さい。

損失と考えられる費用として、不要な開発費用、やり直し費用、バグつぶし費用、再評価費用、市場対応費用、手待ち時間ロス、対策の横展開未実施によるロス、部材の流用・共用不可能によるロス、技術者のレベル向上阻害による長期にわたる人的生産性のロス等数え上げたら切りが有りません。目には見えないかもしれませんが数十パーセントものロスになっているのではないでしょうか。

日本におけるソフトウエア開発企業は今、要件定義・設計図・構造図が書ける技術者育成に投資を惜しむべきではありません。日本で生き残る道は他にはないでしょう。

オフショアによる受注金額・仕事量の大幅減、社内におけるロスの撲滅、新たな付加価値の高い仕事の獲得の三つを同時に解決する方法は要件定義書・設計図・構造図が書ける技術者をどれくらい揃えられるかにかかっていると言っても過言ではないでしょう。

日本の市場ではお客様の言葉をシステムの言葉に変換できる要件開発技術者、要件定義の言葉をさらに開発者が理解できる言葉に変換できるシステムの設計図・構造図が書ける構造設計開発者の需要は限りなく存在するでしょう。



モデル・ベース開発のすすめ

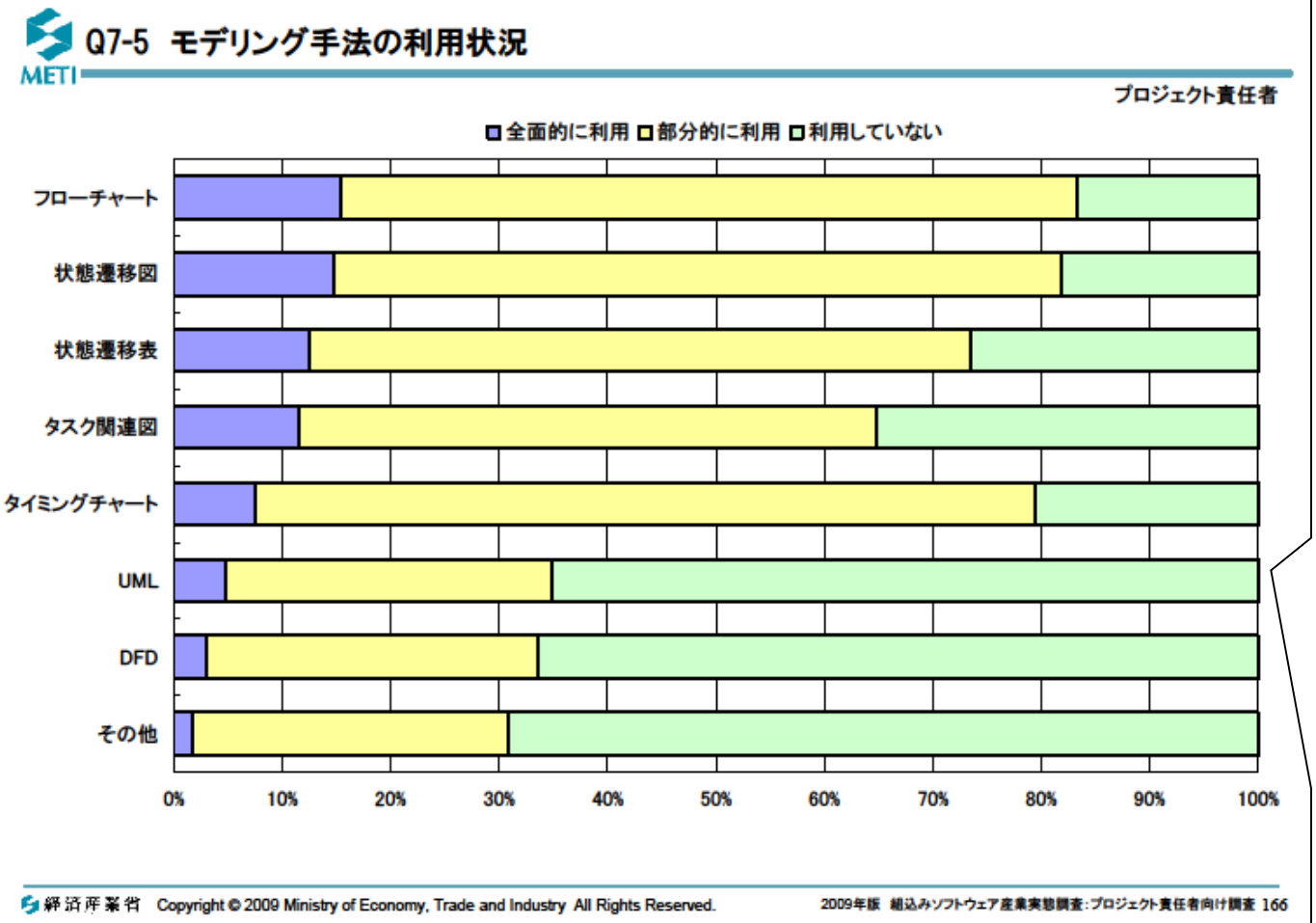
これまでのソースコード中心の開発から設計図中心の開発へ移行する手法としてモデル・ベース開発があります。

従来の手法ではユーザ・要件定義者・PM・開発・評価担当者間のコミュニケーションを行なうための全体を通しての共通言語は存在していません。 そのためシステムの巨大化・複雑化も相まってユーザの要求とおりの製品を実現することは極めて困難です。

モデル・ベース開発は上流工程の要件定義の品質向上に大きく貢献するだけではなく、開発に係わるユーザをはじめとする全関係者を結ぶ共通言語を提供しQCDの全てに貢献するものでしょう。

モデリング手法の利用状況

下記は2009年8月に経済産業省情報処理推進機構より発表された日本の組込みソフトウエア産業におけるモデリング手法の利用状況のデータです。



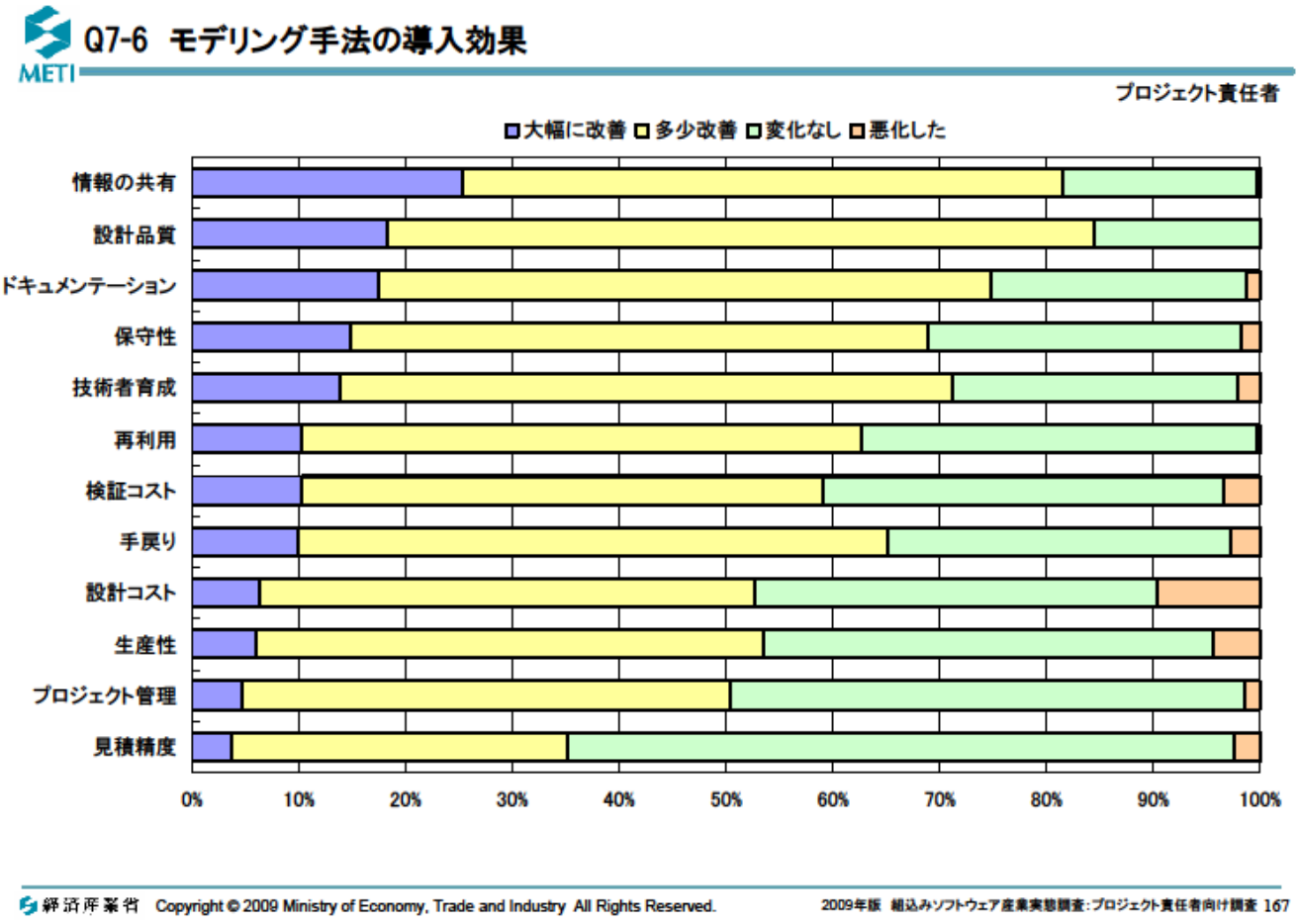
モデリング手法の世界標準であるUMLのプロジェクトでの使用率は、全面的に使用が4．8%、一部使用が30%で合計34．8%とまだまだの状況です。
世界的には90～70%程度のようで、すでに常識の世界のようですが日本はかなり出遅れているようです。

日本で普及しない言い訳
・顧客も勉強してないから顧客との共通言語にならない。
・モデル化・オブジェクト指向の概念の学習がむつかしい。
・設計工数が増える。

＊いつまでもそのようなことばかり言っていられるのでしょうか？

モデリング手法の導入効果

下記は2009年8月に経済産業省情報処理推進機構より発表された日本の組込みソフトウエア産業におけるモデリング手法の導入効果のデータです。



表で明らかなようにモデリング手法の効果が大きいようです。
大幅に改善した項目は以下の通りです（括弧内は“多少改善した”数値を加えたもの）。、

- 情報共有　25．3（81．6）%
- 設計品質　18．2（84．4）%
- ドキュメント　17．5（74．8）%
- 保守性　14．9（68．9）%
- 技術者育成　13．9（71．2）%
- 再利用　10．3（62．6）%
- 検証コスト　10．3（59．1）%
- 手戻り　9．9（65．2）%
- 設計コスト　6．3（52．7）%
- 生産性　6．0（53．5）%
- プロジェクト管理　4．7（50．4）%
- 見積精度　3．6（35．1）%

“多少改善した”を加えれば圧倒的に効果があったとのアンケート回答になっています。

モデリング手法の導入

日本における過去の取組み状況から見て現実的な導入方法は、UMLだけにこだわることなく自社のソフトウエアの性質および技術レベルを見極めて、非UMLのモデルも取り入れた形で、開発の各工程に適切なモデル図を使用することがポイントです。要は現状開発における曖昧さ・不透明さを排除し、お客様と開発が共通の認識をえられる方向性で進むことにあります。

下記に主なモデルの表記法を示しました。

	データ構造のモデル（静的モデル）	振る舞いのモデル（動的モデル）
UML表記	●クラス図 その他に、 〇コンポーネント図 〇配置図 〇オブジェクト図 〇パッケージ図	●ユースケース図 ●シーケンス図 ●アクティビティ図 ●コラボレーション図（相互作用概念図） ●ステートチャート図
非UML表記	●DFD（データフロー図） ●ER図 ●ディシジョンテーブル	●フローチャート ●CFD（制御フロー図）
非UML表記	●画面遷移仕様書 ●性能・消費電力等の非機能要件 ●その他、分野独特の表記法（制御ブロック線図など連続系のモデル表記法） ●形式仕様言語などテキスト言語によるモデル表記法	

出典：日経エレクトロニクス　2006.9.11

業務文書(ビジネス文書)

業務文書には報告書、障害報告書、連絡書、議事録、ビジネスメモ等さまざまな文書があるが、良い文書に出会うことは少ない。むしろ問題解決のために出した文書が問題を更に複雑にしたり、大きくしたりする場合もある。

業務文書の問題事例(訳の分からない報告書)

- そもそも出す相手先が違っている
- 題名と内容の不一致
- 主語がない文章
- 期限が明示されていない
- 結論不明
- 依頼内容不明
- 説明不足
- 事実を隠蔽したまま
- 不正確

文書化の効用の再認識

文書は単なる文字と言う記号の羅列ではない。基本的にビジネス文書であれ私信であれ人が人に対してものごとや心情を伝えるものであると言う原点を忘れてはいけない.

●知識・情報の共有化に有効。

●約束・契約の証拠となる。

●一度に多数の相手に情報を伝達でき、何回も繰り返す必要がなくなる。

●客観的な情報を伝えやすい。

●大量の情報を伝達できる。

●送り手も受け手もお互い相手の時間に縛られない。

●証拠が残り、「言った。言わない」問題がなくなる。

●重要なこととそうでないことの区別・選択がしやすい。

●文章にすることで論理的な思考がきたえられる。

業務(ビジネス)文書の基本要件

内部資料の業務文書意識から対外性を意識したビジネス文書への転換。

■受取る側の利益・心情を優先すること

●まず結論の記述から始めること。

忙しくて時間のない相手はまず結論を知りたいのです。

相手の「何故？」にまず答えて下さい。

●自分の言いたい事を優先しないこと。

まして言い訳から記述を始めないこと。

ビジネス文書に言い訳は不要です。

相手を不愉快にさせるか怒らせるかのどちらかになるでしょう。

業務(ビジネス)文書の基本要件

■正確性を確保すること

●主題名と内容の合致

「名は体を現す」と言われるように、文書の主題名は重要である。

主題名が適切でない場合、これ以降に続く文章自体が誤解の元になる。

主題名は良く考え適切な題名を記述すべきである。

●主語を省略しないこと。誰がやるの？当事者はあなたじゃないの？

主語がないと言うことは責任者等が不明と言うことになりかねない。

●期限を明示すること。・・・いつまでにやるの？

●形容詞・副詞等の情緒的・感覚的・感情的文言や文章は極力避けること。

●事実に基づいた客観的な内容であること。

●数字に基づいた内容であること。

業務(ビジネス)文書の基本要件

■メリハリの利いた文書にするために

●結論に始まり、続いてその経緯について簡略にまとめること。

本文は1～2ページにまとめ、詳細内容は添付資料とする。

●重要なこととそうでないことを区別すること

●同じ内容を形をかえてだらだらと繰り返すような冗長度を避けること

日常文書のコツ

●自分自身の頭で考え、それを率直な言葉で伝えること。

●書き始めるのは、考え方が八割まとまってから。

●考えること、考えていることを言葉に書くこと、実際に行動することはバラバラではいけない。

●書くことはコミット（誓約・約束）することと同義です。

●安全パイの文章ばかりで、相手から「それでどうしたの？」と言われないようにすること。

●文章はなるべく短く簡潔明瞭する。ダラダラと書かない。

●意識的に主語をはっきりさせる。

●タイトルが難しい。何を伝えたいかはタイトルに表す。

●結論を先に示して、理由や詳細は必要に応じて読めるように添える。

出典：「実践！書く力」（プレジデント2007.6.4号）

日常文書のコツ

●書いた文章は必ず一度、読み手の立場になって読み返す。
　読んだ人の反論を想定しながら書く。
●自分と読み手の立場を意識して書く。
●根拠は客観的なデータで裏付ける。
●感情的・情緒的文章を書かない。
　客観的根拠なしに相手を説得しようとすると感情的・情緒的文章に
　なりやすい。
●メールは最も自己中心的になりやすいツールだから、「私」を中心にせず
　「あなた」の立場で書くようにすれば感情を前面に出すことを避けられる。
●ネガティブな球は一度受け止め、ポジティブにして返す。感情の応酬の
　連鎖に陥らないこと。

出典:「実践！書く力」(プレジデント2007.6.4号)

ご静聴ありがとうございました。