社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－SE－153 (6)
2006／7／10

システム／ソフトウエア開発ビジネスの在り方

河野善彌*, 陳慧†, Hassan Abolhassani ‡

この報告は，システムを開発する企業の在り方に焦点を当てる．受注生産型と自主商品型とがある．成功の為には両者共に「広義の技術力」が必要である．ソフトウエア企業の研究開発費率は低く，「広義の技術力」が低い懸念がある．ヒトの知の構造の研究結果によると，製品は小さな段階毎の単位的知識が集合して構成される．企業組織の広義の技術力は，この単位的知識の総集積量で決まる．自主製品型では常に同方向の作業を続けるから，広義の技術力が集積し易い．自主製品型の優れた企業を紹介している．

Business Models of System / Software Development Vendors

Zenya Koono*, Hui Chen † and Hassan Abolhassani ‡

This paper focuses on business models of system development organizations. There are two categories. One is to develop upon order while another is to develop own products. For a success of the business, it is indispensable to own engineering power in the wide meaning of the word. R&D intensity of software organizations is low, and it is feared that their engineering power in the wide meaning of the word is also low. The studies on human knowledge structure revealed that, the accumulation, of small unitary knowledge in the organization, reflects on the product. Thus own engineering power in the wide meaning of the word may be regarded as the total amount of the knowledge. As an organization, supplying own products, experiences efforts in the same direction, they can more easily accumulate the knowledge. An excellent organization supplying own products is introduced.

1．はじめに

ここ数年来，我が国では情報処理システムの大きなトラブルが社会を震撼させた．ついには，情報処理学会の鼎の軽重迄問われた．その後も，組込み系のバグ多発が報じられている．これらは座視できない．

「火の無い所には煙が立たない」と云う諺がある．マスメデイアに報じられるのは，氷山の一角に過ぎない．火消し以上に重要なことは，「氷山」に改善を施して，将来の問題を防止することではなかろうか？

個別仕様で開発されるエンタープライズ系システムの問題が多いことから，前報告[1]では出発点である発注者側に焦点を絞り，仕様を確定させること，更に発注者と受注者間のインタフェイス厳守を提言した．

この報告では，流れで次位のソフトウエア企業に焦点を当てる．一般社会では，注文して作らせる事は当り前ではない．各企業は自主製品で営業することが普通であり，商行為の規範は明確で，市場には競争があるから，営業も含めた総合的技術力（以後『技術』と記す）が企業の栄枯盛衰を決める．

日本では，他国より個別開発システムの割合が高いと云われる．個別に仕様を出して開発する場合には，受注合戦以外は，公にもならず競争も無い．ともすれば規範や規律が失われかねない．現在の問題の背景には，これがあるかと思われる．

更には，競争意識が乏しくなることから，『技術』がないがしろにされていると感じる．これらから，受注して作る側の技術への姿勢を調べた．企業は営利で生きているから，そのビジネスモデルを問うことになる．

２章では，産業界のビジネスモデルとして仕立屋を考え，広義あるいは総合的な『技術』の重要性を示し，３章では売上高研究開発費率の視点からソフトウエア業界を評価する．実にお寒い状態である．４章では知の体系を説明する．これは筆者等のソフトウエア自動設計の研究からの敷延で，知の総量等から広義あるいは総合的な『技術』の様相が判る．５章は自己技術の集積を重ねるパッケージソフトウエア会社を紹介する．「お寒い状態」と「模範例」を対比することで，あるべき姿を皆で考えたい．

* Creation Project koono@vesta.ocn.ne.jp
† 国士館大学 chen@kokushikan.ac.jp
‡ Sharif University of Technology Abolhassani@sharif.edu

2．ソフトウエア産業のビジネスモデル

日本のソフトウエア産業は受注開発に偏っていると云われる．特に米国人にこの意見が強い．筆者等も最終的にはエンタープライズ系の仕事は，5章のような各領域毎のパッケージソフトウエアになると予測する．

自主営業の課題を明確にするべく，洋服製造業を素材にした．既製服はパッケージソフトメーカに，注文して作る「仕立屋さん」が受注形企業に相当する．

イージーオーダは昭和24年(1949)から始まった．少数パターンの既製服と効果な仕立屋さんしか無かった時には，おおいに受けた．今では，あらかたの需要は既製服で賄えるが，イージーオーダの価格は既製服より僅か10～20%しか高くないから約15%のシエアを得て，落着いている．イージーオーダを注文すると，担当者はメジャーを片手に採寸しA4フォーム1枚に寸法等を記載して行く．作業中に「ベンツはどうなさいます？」「要らないのですね」等とオプションを決めさせる．終わるとフォームを見直し，「お客様なら，センターベンツの方がお似合いになりますよ」等と誘導して再調整する．将に要件を採取している．カジュアル～スポーティの数系列を準備し，スタイル毎に「胴体を細目に」等の差異があり，採寸箇所，オプションの種類や選択幅が違う．(メインフレームでの大型システム開発が1段落した後，エンタープライズ系の仕事は定型化されて，「要件を伺って」作る状態であった．これはイージーオーダーに近いのでは？)

本格的な仕立屋になると，採寸の前から違う．「どんな時にお召しになるのですか？」等，話す内に注文主の住む社会／立場等の情報を採取する．目的に応じた系列／スタイル等を推奨して貰え，オプションの幅も広く，細やかな注文に合わせられる．仮縫いで身体に適合させ，趣向に一致するか確認し，細部仕様を調整する．出来上りは，身体によく合い，動き易く着崩れしない等満足度は高い．しかし価格は既製服の約1桁上で，約3%程度のシエアと云う．(大規模ユーザ向けの特注システム開発は，これに近いのでは？)

既製服製造業では「安くする工夫」が大変である．大量製造で素材は安価に購入でき，手順を標準化して技術は高くないが賃金の安い人を雇う．しかし，売れなければ多量の戻り品で倒産する．体形系列や頻度分布が関わり，市場ニーズ，趣向や流行を把握した製品体系をつくる．これらは市場で試行錯誤しながらノウハウを積上げて行く．的中率を高めるには，その会社に高い『技術』が要る．既製服が過半のシエアを獲得できるまでになったのは，イージーオーダーの出発から29年後の昭和45年(1970)．現在のシエア80%強になるには，更に30年を要したと云う．

既製服製造業は組織で『技術』を持つ．仕立屋は『技術』ある人を雇用して『技術』を獲得する．イージーオーダーは中間的な『技術』で成立つ．何れにせよ，『技術』が不可欠である．この『技術』の確立には時間，労力，お金が掛ることも異論無いことと思う．

プログラム作りに対応する実現技術は，ここでは「採寸」「型紙作り」「裁断」「縫製」等である．この上位に方式技術として，

・如何に商品を編成し，各商品を設定するか，
・顧客に訴えるポイントを，価格重点～非価格競争力の何れの位置にとるか，

等があり，その競争力は，関係する実現，方式，営業等の各技術を総合した広義の『技術』である．

前報[1]で異業種のユーザの厳しい言葉を紹介した．

・*建設業は期限を守らず，品質を保証しないことはまずないが，ソフトウエアベンダーでは許されてしまう．*
・工期遵守意識が希薄
　＊工期の遅れ＝ペナルティの意識がない
・設計変更契約の概念が無い

業界の内にも，営業さんの「今回のご注文はこの辺で絞めましょう」，とか"code and fix" 等，洋の東西を問わない自嘲的なささやきがある．勿論，全てのソフトウエア企業がこのような訳ではない．しかし，これらは氷山と云える．一体『技術』はどうなっているのだろうか？

3．ソフトウエア企業の研究開発費率

企業が『技術』を大事にする度合は，売上高研究開発費率（＝研究開発費／売上高，R&D費率）で計られる．総務省の集計結果[19]から抜出した各業界の値を纏めた結果を図1[2]に引用した．図は各業界毎に主要会社群の研究開発費率であり，最高8.4%（医薬品）から最低の1.5%(鉄鋼)と1.1%（食品）の間に分布する．医薬品の率が高いのは，新薬の試験関係の負担が大きいことによる．鉄鋼は成熟期である為，また食品は歴史的に最も古く『技術』確立済みである為に低い．製造業の平均は中間のやや下の3.7%である．

業界毎に主要各社の費率の推移を見ると，高費率ほど，その会社は元気が良い．ソフトウエア業界はどうであろうか？ 情報サービス産業協会(JISA)の基本統計調査[3]を用いて説明する．JISAは日本の情報サービスの主要な企業700社以上で構成された業界団体である．この統計調査には例年過半の会員会社が回答する．

図2の最下部のカーブは日本のソフトウエア業界の研究開発費率の推移[3]を示す．これには教育投資も含むにも関わらず，安定的に僅か1%程度に過ぎない．図3[3]はソフトウエア業界の研究開発費率の分布で，高率の会社も僅かにあるが，1%未満が圧倒的で，費率を増すと急速に減少する．

他国ではどうであろうか？米国では，小さいベンチャー程『技術』開発に新熱心で事業成功を目指すと云う．事業形態が違うが，Microsoftは60%台，Oracleは20%台との報道がある．

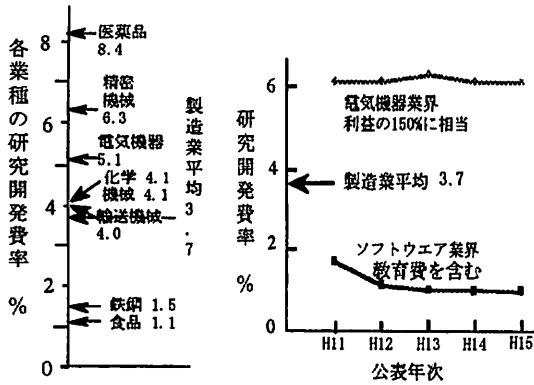

図1 各業種のR&D費率

図2 ソフトウエア業のR&D費率

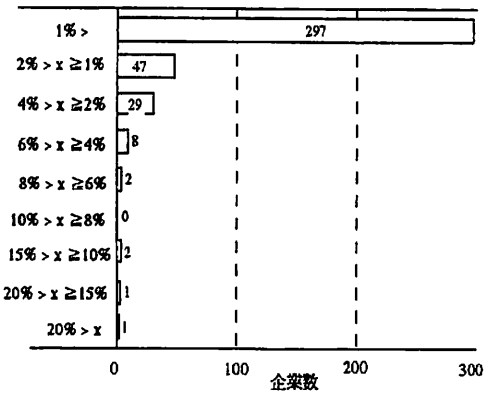

図3 ソフトウエア業のR&D費率の分布

情報系技術に関わりが深い電気機器業界の代表的会社100社の平均値[2]は，図2の上のカーブで安定的に約6%である．この費率は利益の150%相当額と云う．ここでは，激しい競争に勝つために，各社は粗利益の中から利益の1.5倍相当の資金を投じても，新『技術』や新製品を産出せねばならない．

JISA統計[3]でみると，2004年度のソフトウエア企業の(本業の利益を表す)売上高営業利益率は加重平均値で4.5%，また(本業外の金融等の収支を含めた)売上高経常利益率は加重平均値で4.46%であり，この数値は他業界に比べておかしくない．しかし，前記の電気機器業界の状態から見ると，粗利益から，まず第一に利益を確保して研究開発は後回しの感じは否めない．平均として『技術』を重視せず，労力提供モデルで利益を得ていると云えば言い過ぎであろうか？

4．ヒトの知の構造

『技術』を簡単に評価する事は難しい．これを単純化して「知識」あるいは「知」と置換えれば，理解し易くなる．この「知識」あるいは「知」を，筆者等の研究結果から解明してみたい．これを通じて「技術が如何に集積されるか」を，ご理解戴きたく思う．その研究は，「言葉/言語」をヒトの「知」の基本構造に据える．人工知能ではなく「人間知能」である．

我々ホモサピエンスは進化過程の最上位に位する．その力はヒトが築いた文明の力によっており，その根本は「言葉/言語」にあると考えられる．「4，5」から粗筋をご紹介する．

我々の祖先は最終氷河期(約13000年前から24000年前)を生き伸びたが，先駆けて地球上に現れていたネアンデルタール人はこの終り迄に死滅した．彼らの知は未だレベルが低く，小屋は作れず，大集団を構成したり遠距離の旅/交易は出来なかった．知の集積～文明の不足が原因で，彼らは死滅したと考えられている．

ネアンデルタール人化石を調べた結果，我々の脳の重量は彼らより平均して小さいが，我々はより自由に豊富な音声を操る身体構造を持つている．具体的には，

・口蓋の空きスペースがより大きく
・声帯がより上に位置している

ことが鍵であったと云う．

動物でも最高数十種の符牒を持ち，相互に情報を伝える種があると云う．ホモサピエンスは，触ったり見たり聞いたり，経験する全てに名を与える．言葉/言語を使うことで，単なる符牒の授受よりは，遥かに高度に相互に意図，指示，情報，感情等を伝えあい，より高効率な集団を形成できる．更には，見えない，聞こえない，あるいは実体の無い抽象的な「美」，「神」や「神秘」にも名を与える．これにより知的能力をより高度に伸ばし，急速に知の成果を蓄積して文明を拓いた，と考えられる．

言葉によれば，複数概念からなる新概念(抽象化)に名を与え，またある名を使えばその下位の概念群を呼出せる．ヒトは，この両方向性から階層的な知の体系を作り，自由に昇り降りできる記憶の階層的な構造を得たのであろう．

これを知の基本構造とすれば，

・ヒトの連想結果が意味ネットワークをなす，
・ヒトの知識の大部分は階層形記憶で，
・エピソード形記憶は少数である，

等の経験的事実を理論付けられる．

筆者等はソフトウエア自動設計を研究する内に，階層的な知の構造が，単なる設計をヒトに倣って設計を可能にするのみでなく，より一般的なヒトの「意図的行動」を可能にすることに気付いた．

初めに設計を説明する．図4は「時計」[6]プログラムの具体化過程を示す．左端のデータフロー図群は，仕様である概念「時計」の階層的な展開連鎖で示す．3段目で入出力データを付加された「時計」の単位データフ

ローは,(MyersのSTS分割により) 直列的に詳細化された3本の単位データフローに展開される. これは親概念から子概念群への展開である. (この並列展開はJackson法の特徴的な展開である.)続けて第4段では, 並列的に3展開している. 以後, 展開毎に概念は明確化し具体化し詳細化される[7]. これは機能のみを考えた構造化設計を超えるもので, Data Oriented Approachの原理にもなる. 図4の右下の階層展開網モデル[8]はこれを示す模式図 (モデル) で, ○は情報, ●は階層展開する処理である. 設計は, この網を一筆書き状に階層展開を繰返しながら降って行く動作である.

各展開毎に,(第3段の左上の開始記号から始まったような) 小さなフローチャートが生じる. 図の中央はこれを示す. これらは接続用で最終段階では消えてしまう. 最下段の「分針の角度を求める」等では, 機能は展開されて単位的処理になっているから, この処理を実現手段であるプログラム言語表記(群)に展開する. これらのコードは図の最右端に示されている. プログラムはあくまで実現手段に過ぎない.

他の実現手段による (例:ハードウエア) 設計も, 階層展開による詳細化を使い, 基本的に図4と同様になる. 設計の本質は言語の概念展開連鎖にある.

この階層展開できる力が「知」の中心なのだから, プログラム/システムを作る『技術』の中核は, 図4の左端のデータフロー群に見られる (ここでは自然言語での) 展開の知識である. コードあるいはプログラムは, 実現手段に過ぎず, その実用上の価値は大きいが本質ではない.

最上位では, 軍事科学での「目的の階層性」[9]が使えることを前報で説明した. 即ち戦争/経営の最終目的が最上級者に課される. 彼はこの「目的」を達成する手段群に階層展開して各手段を夫々の部下に課する. 各部下も同様に繰返し展開する. 戦争/経営の計画は基本に於て設計と同様な階層展開連鎖状に, 明確化し具体化し詳細化する過程である.

最上級者の展開結果は戦略, その後の具体的な展開段階を戦術と云う. 最上位の「戦略の失敗は (後続段階の) 戦術では取り返せない」と云われる. 最大の力は最上位にある.

ヒトの肉体的動作でも同様である. ある上位概念 (例:「写真を撮る」) を与えると, これを階層展開(「カメラを取出す」「対象を狙う」「シャッターを押す」) してより詳細化し, 必要なら更に階層展開を続けて, 筋肉組織を作動させる神経指令に落として, 肉体的動作を行わせる. (最後の指令をアクチュエータに与えれば人間ロボット (ヒューマノイド) が実現できる. ここではプログラムは必要ない.)

これらは全てに共通な「意図的行動」である. 従来プログラム中心に考えて,「プログラム / ソフトウエアは他とは異なる」と思われてきた. しかし,

<u>ヒトの意図的行動と云う共通視点から見ると, 共通性に立って, 他～先行例に学ぶ / 倣う</u>

ことが重要である. 主要な例を幾つか示す.

図5[10]は産業界一般に云う所謂「工程 (作業工程)」と組合せた図で, 最下部には水平方向の階層展開網があり, これは設計の概念展開連鎖(product)である. 最上位は「開発」工程, これを「設計」工程と「テスト」工程に展開し, 設計は更に (図4と同様に) データフロー設計.... と展開される. 展開するにつれて, 如何なる作業を行うのか, 具体化し明確化し詳細化し, 最後はヒトの単位的処理指令になり, 最下部の設計展開のある要素を制御する. 工程の階層展開の体系(process)は設計展開とは相互に独立な知識体系になる. 「工程」とは作業対象や作業方法から独立した共通技術である. そこで,

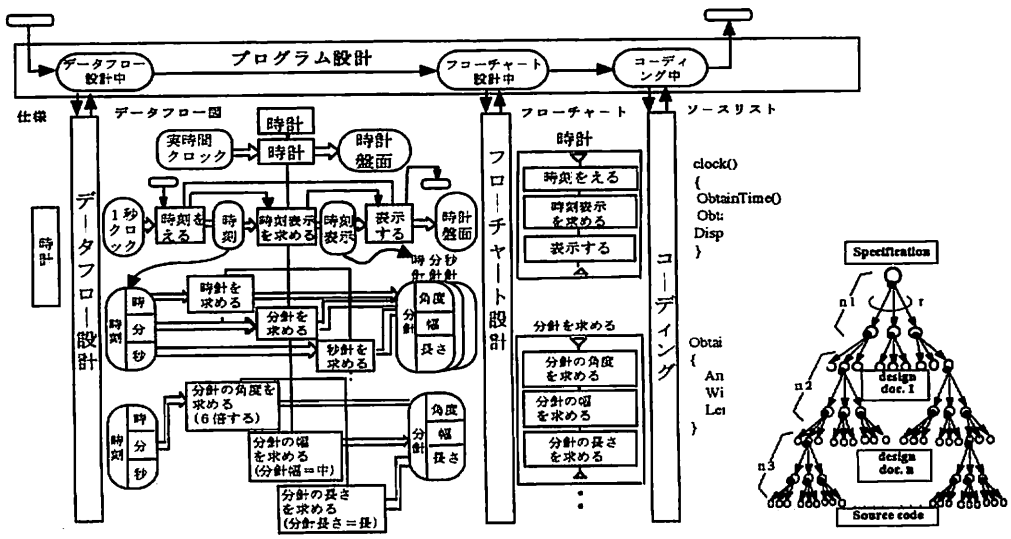

図 4 時計プログラムの設計軌跡

産業界で用いる「工程／作業工程」とその技術は，システム／ソフトウエアに適用できる，逆にシステム／ソフトウエアのprocessについて，一般的な「工程／作業工程」とその技術に反するものは，中核ではありえない．

世界に稀な「日本のソフトウエア工場」[11]あるいはメインフレーム系ソフトウエア開発組織は，この「工程／作業工程」を1960年代から使用している．これは「意図的行動」の拡張の第２である．

他の拡張を説明する．図4の左端のデータフローを見る．この展開を○○システムの展開に適用すれば，この下は，○□サブシステム，○△サブシステム，○▽サブシステムのようなシステム/ソフトウエアの階層的な機能あるいはプログラムの構成になる．機能とプログラムの構造は，概念の階層展開連鎖である知の構造に対応することが判る．これは第３の拡張である．

製品の構成と開発／作業する組織の構造とは主要部分を一致させることが多い．例えば最上流部分は○○SE（System Engineering）チーム，次に展開して各サブシステムに対応して入力処理チーム，内部処理チーム，出力処理チーム等にする．「工程」の流れに沿った組織が対応する場合があり，最上位は設計部長，データフロー設計は方式設計チーム，フローチャート設計は詳細設計チーム，コード化はプログラミングチームのようになる．これらproductやprocessの階層構成に合わせて組織が作られることは，働く者にとって判り易い（即ち頭脳に負担を掛けない）良い区分である．組織の構造への対応が第４の拡張であり，古代からの軍や教会の組織もこれで説明できる．これら階層性に異論を唱える向きがある．論拠は局所最適化で，今議論している大局的最適化のレベルではなく，説得力が無い．

次に，知の働きの様相を説明する．今迄，単純化して「親と子概念群の対」で説明したが，この展開対を（例えば言語規則等の）ルールで作ることもでき，また基礎的概念（深い知識）から創りだすこともできる[14]．また親から子への展開には認知，候補選定／判断等がある．全てに共通な様相が現れるが，以後は理解の便宜上簡単な展開対の場合を説明する．

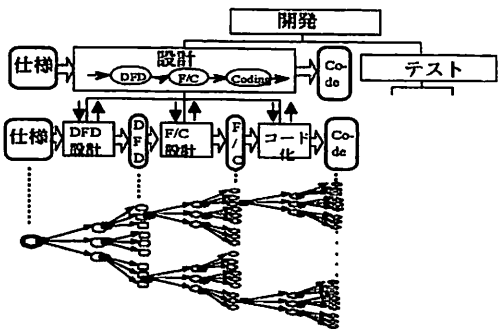

図5 工程の構造

ヒトの頭脳では，各展開対はばらばらに記憶されている．しかし，ある概念が与えられると，次々と展開連鎖を構成して，ある意図的行動をする．元は頭脳の「記憶」であっても，階層展開連鎖を紙に記録できるし，展開中に既存の記録を使っても（所謂再利用）構わない．

同様に，（図４のような階層展開連鎖でなくても）設計文書も同様な知の体系である．殆どの設計文書作成テンプレートは階層的に記述するように誘導するし，ベテランは，これら無しでも，階層的に記述し，構成する．

階層的に展開したデータフロー群は，小さな進行段階毎だから展開毎に正しさをチェックすることが容易である．設計文書をこまめに残してチェックすれば（これは理数系の試験答案の作成と提出前のチェックと同じ），ソースコードに伝搬する前に大部分の誤りを消せる．（日本ソフトウエアの品質の良い原因はここにある．）

この知の集積は如何に形成されるのだろうか？筆者等は定量的に研究した[12, 13, 14]．図４の階層構成のデータフロー群の機能部分が構造化チャート（PAD）では樹枝状展開になることを用いた．図６[12]は実験結果で，横軸は設計経験量を最初のプログラムから出発した維持管理（変更／拡張）プログラム設計の回数である．a図は設計経験と共に新規な展開対が蓄積する様相を示し，グラフは初めに急激に立上がるが，次第に勾配が緩やかになる．これはヒトが新しくゲームを行う場合の様相（習熟効果）に似ている．

b図は両対数尺度での表示で，プロットは直線状の傾向線を示す．ハードウエア製造の広義の生産管理Industraial Engineering (IE, 経営工学)では作業者の作業効率を研究し，習熟効果を見出して定式化し，技術を確立した[13]．この場合は対数習熟効果と云う．（ヒトや組織の習熟特性は殆ど全て対数習熟効果である．）この結果は重要な意味を持つから，再度確認を行なった．第１と第２の特性は同勾配で，対数習熟効果に間違いない．

直線傾向線の勾配式を用い，自動設計の単位展開当りの平均設計時間を計算した結果をc図に示す．経験数が増す程，知識ベースに蓄えられた知識量が増すから展開時間は小さくなる．過去の同種の経験を積むにつれて，初めは急速に速度が向上するが，次第にその増加は緩やかになる．しかし，何時までも向上を続けるヒトの対数習熟特性を表す．（図のカーブは初期段階を示しているが，経験数が多くなれば連続的カーブと見做せる．）

この実験は，ヒトの習熟効果は知識の対数習熟状の蓄積によることを，理論モデルを用いて定量的に示した世界最初の報告である．この他のルールレベルや基礎知識を用いる場合等も実験した．全ての知識の量には経験回数につれ対数習熟状に蓄積される傾向がある[14]．これはヒトの知識一般の対数習熟状の効果を定量的に示した世界最初の報告である．

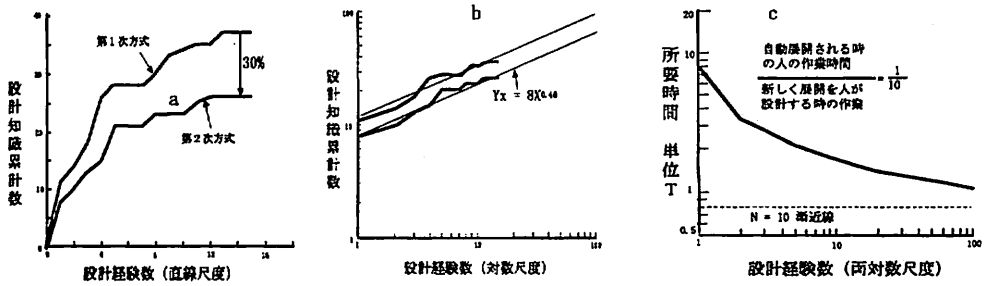

図6　設計作業の習熟効果

ここでは予め初回の母胎を作り，実際の開発作業と同様な所謂保守（厳密には維持管理）を行わせた．この実験は1回の出現で完璧に記憶する方式である．機械や動物の刷込みと違って，人間の記憶はなかなか定着しない．また集積量を増すには時間が掛る．ある領域でエキスパートに成るに約10年を要すると云われる．ヒトの社会では，若年者には教科書を与えて学ばせる．成人の新参者にはマニュアルを与えて，基本パターンを倣わせる．また，各種の標準や仕掛けを準備して，少々ヘマをしても他がカバーしあう．これらから，知の集積期間の長さは一見あまり目立たない．

- 独自『技術』の立上は長期間を要する
- 先行 / 他の例に学び / 倣えばより早くできる
- N段に分業 / 専業すれば総期間は1 / N に減る
  フォードの創始した流れ作業はその好例．
  SE，設計，プログラミングの3部隊に3分割し仕事を流し続ければ，1/3の期間で同程度に習熟する．
- 一開発でもN 回繰返しで作れば，より効率的
  Rational Unified Pocess はこの方式の例
- 開発要員の作業対象がガラガラと変ると設計の知は集積されない
  米国人プログラマは駆使できる言語種数で給与が変る．アプリケーションには関心が低く，深入を嫌う．新言語（プログラミング）が学べる職場の求人に応じ，習得するとより高い給与を求めて退職する．
  （内部昇進できる日本では）ある領域に専従し続ける人は当該領域のエキスパートに成長する．しかし，エンタープライズ系で働く日本のソフトウエア人は共通技能であるスキルに熱心であるが，より本質的な方式技術は尊重しない．それは方式技術の深さを経験できない立場に陥れた結果と思われる．
- チーム編成をガラガラと変えたら，工程の知は集積されない．
  産業界ではprocessは人にありと考え，それを文書化する以上に，従事者達を固定し，専業化させる．設計と製造の分離はその例．エンタープライズ系の工程特性はバラツキが大きい特殊性があるが，それは頻繁に人を移し替ることも影響すると思われる．

5．OBC社のビジネスモデル

研究開発重視型の好例として，基幹系ソフトウエアパッケージを提供するオービックビジネスコンサルタント社（略称：OBC社と云う．）を取上げる．既存公開資料[15，16]のみでは定量的資料が不充分なので，投資家向の公開資料類[17，18]の提供をお願いして調査した．以下はその纏めで，総括した公開を許諾戴いた．

OBC社は昭和55年(1979)にコンピュータ販売，プログラム販売および設計業務を目的として設立され，翌年より現社名になり，初期製品「TOPシリーズ」の販売を開始した．平成5年(1986)には「奉行シリーズ」の発売を行い，現在では東証一部に上場している．表はパッケージで提供する処理機能を示す．

1995年以降の(非連結)売上高と本業の利益である営業利益を図7[18]に示す．売上の源となるユーザは現在は約40万社に達し，販売店は約3000と云う．これを支える（同社単体の）人員数は現在約500名に過ぎず，内約1/2が技術者である．

2006年3月期の営業利益率は38.4%で，JISA統計によるソフトウエア企業より約1桁近く高い．図7の右下の図[18]は，近年の研究開発費率を示す．研究開発比率は2000年の約5%から増加しつつ2006年は10%に近い．これも約1桁近く高い．

売上の構成は下記3項で，各各は約1/3づつである．

1．パッケージソフト類（ソリューション），
2．指導や保守（サービス），および
3．関連製品販売

即ち，全ての機会を活用し収益を上げている．手間の掛る売込/入門～導入～運用等は，要員への教育を徹底させた上で販売パートナー各社が行ない，かくして顧客の高い満足度を獲得している．知識集約型企業を標榜し，自社の営業要員は全て日商簿記2級およびマイクロソフト認定技術者の資格を持たせ「システムコンサルタント」にする．（分業 / 専業化を活かしている）

経営戦略の源は創立社長 和田成史夫妻(共に公認会計士)から始まる．創立期の中心指導者は社長中心の公認会計士/税理士達であった．これは戦略的な経営に反映している．（上位の知はアプリケーション重点）

表　製品パッケージ名称

区分	名称
財務会計システム	勘定奉行
	勘定奉行（個別原価管理編）
	建設奉行
給与勤怠人事システム	給与奉行
	就業奉行
	人事奉行
	法定調書奉行
販売・仕入在庫管理システム	商奉行
	蔵奉行
固定資産管理システム	償却奉行
顧客管理システム	顧客奉行
申告書作成システム	申告奉行（法人税・地方税編）
	申告奉行（内訳書・概況書編）
	申告奉行（所得税編）

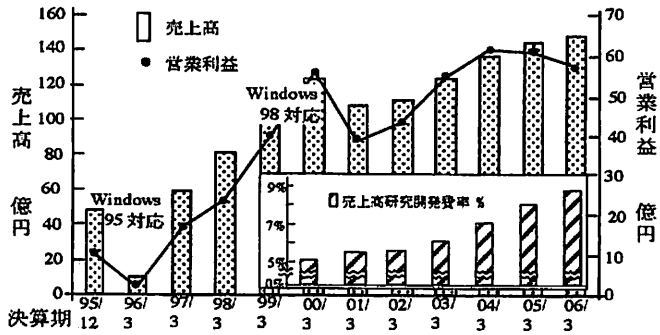

図7　売上高，営業利益と研究開発費率の推移

ソフトウエア開発は，上記の人達の中でソフトウエアに明るい人々が，若者達を教育しつつアプリケーションを作らせ，ソフトウエア設計者を育成した．事業創始期の苦い経験を活かし，

「自由と公平」「採用と教育」「革新と職務」

を基本方針とした．ある時期から，重要な技術変革は，自社採用で育った若手の中核人物に集中検討させて，新方向を打出して推進している．（知の独自開発）

パッケージソフトウエア等，専用的なシステムを中心として営業する時の開発作業は所謂エンタープライズ系の受注開発とは大きく異なる．若干を解説する．新しいアプリケーションを開発する時，充分に注意しても初回は問題が多く遅延や品質不良に陥る．しかし，この母胎とそのチームの陳容を護りつつ，維持拡張作業（平均約に約10%／年の作成規模）を行えば，格段に問題は減り，生産性や欠陥率は向上し，バラツキも減る．（これは初期は習熟効果が大きいことによる．）実績生産性や欠陥率に準拠して次の開発計画を作り計画すれば再現性は高く，念入りに管理すればバラツキは±20%以下に収められる．この進歩は習熟効果，即ち知／技術の集積によるもので，初期の数回は顕著に向上する．しかし，向上の度合いは徐々に低下して行くから，ローテーションする等して向上率を保たせる．全員の技術が上がり，特性が安定になるとハードウエアの設計や製造の場合と同様相になり，定量的な工数や欠陥の実績から工程の問題点が割出せる．これらの改善施策を実行すれば，計画に近い改善が達成できる．母胎を護り，これをベースに各種拡張を行えば急速な展開も可能になる．このように，対象／顧客／リーダ／メンバが全てガラガラ変り，その結果として各種の特性が大きくバラつく場合とは全く違った様相になる．

事業展開は基本方針とおりに戦略的である．製品計画は，ローエンドは製品レベルを細かく設定するが，個別相違が大きな（大規模～一部の中規模）会社には初めから対応しない．使用環境への適合方式を変える（例：LAN利用）ことで規模上限を上げる．初期はMS-DOSで8/16bit MPUとBASICから始め，Windows 95以降は32bit CとVC++と，作業対象と技術を同期的に上げてきた．これは技術者達の実力向上と人員増強過程でもある．

常に顧客の要求に耳を傾け製品化する．他社システムとデータベースでインタフェイス可能にして，逆に自社領域を確実にしている．また，各銀行システムに対応可能である．カスタマイズの要求に応えつつ，その中から有用なパターンを拾い上げコンポーネントと名付けてオプション化する．業種毎に異なる処理等はテンプレートとしてオプション化して，全体として纏まったトータルソリューションを提案可能にする．2005年末の雑誌調査では，高い顧客満足度が定量的に示された．（これらは産業界での模範的行動と思われる．「その場毎に個別調整する」取組みではこのような知は集積されない．）

Microsoft社とも親密な関係を結んでいる．次期システムとして，2003年から.NETに合せて64bit MPUとC#への切替を準備してきた．β2版の開発も済ませ，Windows Vistaのリリース開始と同時の新製品発売を計画している．巨大な中國市場では30%を占めたいと，中國要員の多数雇用を最近発表した．同時に従来の基幹系から情報系への進出も目指している．

成田社長は2005年中間決算の報告を「...社員の成長のステップに合わせて，人間の本質を探究する研修を行っています．」と結んでいる．各種経験が統合されて最も重要な人の育成に帰着している様子が伺える．

組織の技術力は構成員の能力向上で達成される．事業は人，人を大事にする組織に実力が付く．

同社のビジネスモデルは基本的に知にある．常識以上に広い知／『技術』の集積が利益を産む力になっている．

6．むすび

この報告では「受注依存型」企業と（OBC社を実例としてお借りして）「自主商品開拓型」企業の2極を対比し，理解し易いようした．平均的ソフトウエア企業をかなり露骨に悪者に仕立てた．それは対極的な姿を示す為とご理解戴きたい．

組織の知あるいは『技術』の総量に対数的に比例して右に伸びる軸を考える．今，前者は左の何処か，後者は右の何処か，に位置していると云えよう．四半世紀前，両者の位置はさほど違わなかった．この四半世紀に前者

は余り進まず，後者は確実に右方に伸びた．各企業は，両者の間の何処かの座標点に位置付けられる．

エンタープライズ系の平均の姿のみ取上げ悪者と決めつけたが，それは説明の便で真意ではない．小～中規模企業の中にも，高度なサービスや製品の提供を目指して地道に努力を積上げ続けて居られる企業もある．OBC社は約四半世紀の耐えざる努力を積重ねて現在の座を築かれた．継続は力である．前記の軸上でOBC社の背を見ながら，夫々の道を拓きいて第２第３の模範生が登場することを願っている．読者の方々が，更なる向上のヒントを本報告の中から読取って戴ければ幸いである．

ソフトウエア企業の経営者/管理者/技術者の方々には，本報告は不愉快なことの羅列であるかも知れない．しかし，エンタープライズ系のトラブルが現在の社会の耳目を集めている．それはソフトウエア企業の在り方の反映だから，トラブルを減らすには，それを変えねばならないのではないかと筆者等は感じる．具体的な事例として，企業の在り方に直結する研究開発比率やパッケージソフトウエアの模範例を纏めた．

現在では，問題はエンタープライズ系から組込み系にも広がっている．これはソフトウエアエンジニアリングコミュニティに投げられた問題ではなかろうか？筆者等は，トラブルの根底/基礎を捕えようとしている．

・何が問題を起こしているのか？

・では，有効な対策は何なのか？

そこまで至らずとも下記の議論もあり得る．

・エンタープライズ系企業はパッケージソフト企業に対抗して，如何なる方策で進めるべきなのか？

これを契機に，皆様方の議論が起れば幸いである．

謝辞

この報告は筆者等が埼玉大学で推進したSoftware Creation Projectの成果を発展させたものである．Projectの力になって下さった学生諸君およびB. H. Far先生(現カルガリー大学助教授)に，ご貢献を感謝します．また，OBC社の資料については高橋知久氏のご協力に感謝します．４章のネアンデルタール人の引用総括については，編者赤澤威先生（高知工科大学教授）および[5]の１０章「言葉を話したか」を執筆された内田伸子先生（お茶の水大学教授）のご指導に感謝します．

参考文献

[1] 河野，陳，Abolhassani，ソフトウエア開発工程の在り方，情処研報，2006 SE-152(2), pp. 9-16, 2006. 5.

[2] 株式会社 東レ経営研究所，平成１６年度産業技術調査 研究開発税制に係る企業ニーズ把握の為の調査，調査報告書，経済産業省，2005年 3月．

[3] 情報サービス産業協会，基本統計2005・概要編の主な数値，情報サービス産業協会，2005年12月．
URL: http://www.jisa.or.jp/statistics/index.html

[4] Sringer, C. and Gamble, C., In search of the Neanderthals - Solving the puzzl of human origin, Thames and Hudson Ld., London, 1993. 訳 河合信和，ネアンデルタール人とは誰か，朝日選書 ５７６，朝日新聞，1997.

[5] 赤澤威編，ネアンデルタール人の正体，朝日選書 ７６９，朝日新聞，2005.

[6] Koono, Z., Ashihara, K. and Soga, M., Structural way of thinking as applied to development, IEEE COMSOC Global Telecommunications Conference 1987, pp. 26.6.1-6, 1987.

[7] 陳 慧，Far B. H.，河野 善彌，ソフトウエア自動設計における系統的なエキスパートシステムの構築，A設計工程からの設計知識の獲得と再現，人工知能学会誌，Vol. 12, No. 4, pp. 616-626, 1997年7月．

[8] Koono Z., Chen H. and Far B. H., Expert's Knowledge Structure Explains Software Engineering, Proc. of Joint Conference on Knowledge-Based Software Engineering 1996, pp. 193-197, Sept. 1996.

[9] Cluasewitz, Karl von, Vom Kriege, 1832. 淡徳三郎(訳)，戦争論，徳間書房，1965.

[10] 河野善彌，陳慧，人の設計知識構造と定量評価(2/2)，信学技報，KBSE2003-58, pp. 73-78, Mar. 2004.

[11] Cusumano, M. A., Japan's software factory: A challenge to U. S. management, Oxford University Press, 1991.
(訳) 富沢宏之，藤井留美，日本のソフトウエア戦略－アメリカ志気経営への挑戦，三田出版会，1993.

[12] Chen H., Tsutsumi N., Takano H. and Koono Z., Software Creation: An Intelligent CASE Tool Featuring Automatic Design for Structured Programming, JIEICE Vol. E81-D, No. 12. pp. 1439-1449, Dec. 1998.

[13] Salvendi, G. ed.: Handbook of Industrial Engineering", John Wiley and sons, 1982. (訳)

[14] Hassan Abolhassani, 河野善彌，ソフトウエアクリエーション:ルールによる自動設計と知識による自動設計，情処研報，ソフトウエア工学138-15, pp.105-112, July 2002.

[15] 和田成史，井田純一郎，加藤雄一，成長と革新の企業経営（社長が語る 学生へのメッセージ），丸善出版事業部，2004. 12

[16] 野村証券株式会社，株式会社 オービックビジネスコンサルタント，先駆者達の大地，IRマガジン2004年春号 Vol. 65，野村インベスター・リレーションズ．
man@bow, URL: http://manabow.com/pioneer/obc/index.html
富士山マガジン，URL: http://www.net-ir.ne.jp/ir_magagine/pioneer/vol065_4733.html

[17] オービックビジネスコンサルタント編，OBCを知るための基礎資料，同社 IR資料，2006年4月．

[18] オービックビジネスコンサルタント編，2006年3月期説明会資料，同社 IR資料，2006年4月．

[19] 総務省，平成１６年度版総務省統計局化学技術研究調査報告，総務省統計局．