Vol.2011-SE-171 No.22
2011/3/15

ソフトはハードと何が如何に違うのか？（１．工程/プロセス）

河野　善彌[†]　　陳　慧[††]

この論文はソフトウエアとハードウエアの研究/開発/生産のプロセス中心の比較論である．通常のハードは自由競争市場向けであり，所謂ソフトウエアエンジニアリングで云うソフトは受託開発向けのソフトウエアである．ソフトとハードを同一基盤で比べると，両者の本質は同様である．従って問題は，自由競争市場向けのハードとソフト対受託開発向けのソフトウエアの差異に帰する．

キーワード：　ソフトウエア，ハードウエア，プロセス，工程，意図的行動

The Vital Difference of Software from Hardware (1. Process)

Zenya Koono[†]　and Hui Chen[††]

This paper discusses differences of software process and hardware process putting emphasis on research, design and development. Design of hardware and original software are for free and competing market, while that of software by s/w venders is custom design, they are quite different with each other. An initial design down to testing is similar. Production of software is just copying of a file to medias, whereas that of hardware is N-time production of the product. Therefore, human resources, investments and equipments are poured into a production plant.

Keyword: Software, Hardware, process, intentional activity

１．はじめに

この論文の目的はソフトウエアとハードウエアの工程/プロセスを対比して差異を明確にして，それらをより良く理解することである．ハードウエアの工程が明確化したのは，19 世紀末から 20 世紀初頭であり，その歴史は 100 年を超える．この工程/プロセスを確立したのは，科学的管理を説いた F. W. Taylor 等による Industrial Engineering, IE，経営工学であり，ハードウエアの大量生産が確立され，今日の隆盛がある．

ソフトウエアが発祥したのは，1946 年に世界最初の計算機 ENIAC が弾道計算用に動き始めた以後である．1951 年には Remington-Rand が商用機 UNIVAC Ⅰを世に送り，1964 年には IBM が System/360 を他社に先駆けて送り出した．かくしてメインフレームが主流になり，1970 年代初頭以来，初めは 4 bit 次いで 8 bit，から出発した LSI を用いるマイクロプロセッサが成長し主流になった．計算機も 50 年余りの歴史を重ねた．ソフトウエアも大発展をとげ，現在の社会を支えている．しかし，どうも今のソフトウエアは元気がないように見える．原点に立ち戻って，生産の流れから再吟味を始める．

２．基本的なプロセス/工程

初めにプロセスの基本形を確認する．図１．a[1][2]は所謂「構造化設計」に従い，機能「時計」につき階層的展開を２回繰返した機能階層図である．図 1．b[1][2]は，遥かに上流の国家的レベルでの機能の階層的展開例で，ある国の戦

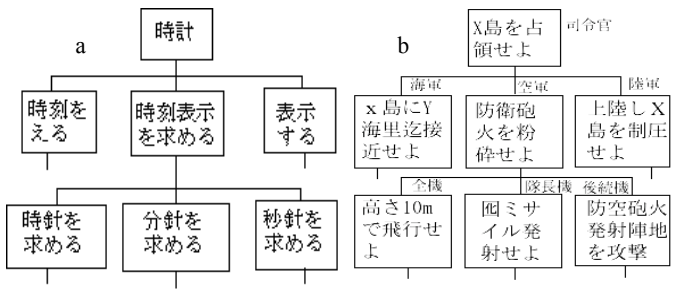


図１　各種の機能階層図

† クリエーション プロジェクト，神奈川県藤沢市本藤沢 2-13-5，電話/FAX：0466-81-6872
Creation Project　Email: koono@vesta.ocn.ne.jp.　URL: http://www.creationproj.org
†† 国士舘大学 情報科学センタ，世田谷区，東京都．
Center for Information Science, Kokushikan University, Setagaya, Tokyo.

ⓒ 2011 Information Processing Society of Japan

争の最終目的「X 島を占領せよ」を，その実現手段群に階層展開して詳細化したもの[2]である．これらの階層展開は幾らでも続けられる．例えば経営上のある最終目的から出発して「製品企画」のレベルに至り，更に詳細化して図１.ｂの「時計」至ったと考えよう．

図２[1][2]は筆者等が提唱する設計で「拡張」構造化設計と名付けよう．図の最上段は仕様である「時計」，次の第２段では，この「時計」の機能箱の左に入力（データ），右に出力（データ）を置いて「時計の単位データフロー（図）」にする．第3 段では前記の単位データフローの左から右へのデータの流れを引き伸ばし，中間の２ノード点[3]でデータを当て嵌め，［１秒クロック→時刻→時刻表示→時計盤面］のデータ変換の流れを意識する．ノード点のデータを決めてから，各データの中間に「機能」を記入すると図２の第３段になる．このデータフローは第２段の単位データフロー「時計」を厳密に階層展開したもので，第２段の時計の実現方式（所謂処理）を示す．単位データフローの中間の機能箱には，これら処理で果たされる「機能」[4]を記入する．これらの図は人の概念操作を正確に記述するもので，ハードやソフトを扱える．各単位データフローは科学技術用語を含む自然言語で表した概念である．特にソフトを意識するなら，階層展開の度に第２段の各機能箱に出入りする矢印で示した制御用のフローチャートが生じる．

図２の第３段に進む．ここでは先に決めた時刻や時刻表示は時，分，秒に階層的に展開されて(先の例では直列３分断であったが)，ここでは並列に３展開している．このパターンは M. A. Jackson がプログラム設計の原理[5] の中で度々指摘している．

第４段は入力側では時刻の「分」であり，出力側では時刻表示の「分針」である．60 進法の分針の角度は，60 進法の分の値を６倍すれば良いから，プログラム設計ならソースコード表記し，例えばハード論理なら分の信号線を×６のマルチプライヤー/乗算回路に入力し，その出力を分針の角度レジスタの入力に繋ぐ．これらから次の性質[1][2]が見える．

１．（設定した出発時の）目的概念から出発し，（科学技術用語を含む，自然言語の）階層展開を行い（図１や２は確定結果を示し，展開の度に複数の子概念群中から選定して使う），子概念群の各１について順次に階層展開を繰返す．繰返すと，対象の概念は，明確化し，具体化し，詳細化し，最後には瑣末化する．言葉で表す概念は，経営でも，システム/ソフト/ハード等何でもよい．

２．この詳細化の流れの中の適当な段階で（例えばプログラムなら，機能に対応するプログラム言語の宣言群（コード）への対応が透けて見える時），次の実現手段（コード）に置換する．「実現手段」がある条件を要するなら，上位から該実現手段に整合するように，各親一子階層展開の度に次第に，にじり寄り（それでも駄目なら別情報を加えて），次の実現手段に置換する．

３．a 前記 1 および 2 項は，ヒトが目的からその実現手段群へと階層展開する意図的行動の習性のモデル化であり，言語で表現される対象一般に成立つ．従って概念展開が設計等の意図的行動の中心であり，ソフト／ハード等は実現手段に過ぎない．ソフトウエアは他とは異なることは，本質としては有り得ない．

３．b 上流での視点．IT 系システムならば，最上位での岐路は投資先で，・IT 系ソフト，・製造設備投資，・資金運用等の選定である．ここでは経営エキスパートの知識が要であり，経営者に経営エキスパートがお話して正しい決定に導くことが不可欠である．更に先に進み具体的なシステムの企画／計画なら，先に経営エキスパートが経営者にお話して決めた方向付けに従い，システムの構築を詳細化していくシステムエンジニアの知識が必要である．経営意図が如何に構成に反映するか，その様相を顧客側経営者とシステム構築責任者にお話して，相互に理解しあえるアーキテクチャが定まる．これは通常一般的な常識である．

４．以後は次々と実務的に展開し詳細化する作業が続く．この間に精密な予測（ソフトなら，ソフト規模，必要メモリ～ファイル規模，処理能力と応答時間等）と評価が必要で，これらで破綻の無い設計が続く．最終段階では対象業務毎に例えば CAD や DA (Design Automation)等の機械的手段を用い，極力誤りを減らし，検証や確認を正確化し容易化する支援システムを使うことが望ましい．

５．この流れのある段階で，システム（組込システムも含み）/ハード側は「これから先は，自社内で作業しますから御一任ください．（詳細は私達に一任して自由度を大きくして頂いた方が経済的です．）詳細は後でお出しする承認図でごらんください」と宣言して引込む．再度打ち合わせるのは試験の前である．

これは洋服の仕立て，プレハブメーカによる建築，ハード装置～プラント等の発注等，殆どの商取引の基本形である．この交渉段階で誠意を持って対応し，投げかけられた諸問題に的確に正確に回答していれば，多くの人は「あの会社はしっかりしている」，あるいは「技術力がある」等と評価する．発注者はしっかりした技術があると見込んで注文する．しかし，「動かないコンピュータ」等の報道から見ると，受託ソフト開発では旨く機能していないようにみえる．要求工学以前に，社会通念での仕様打合わせの方式を徹底させるべきではないのか？

2 これはプロシアの軍事哲学者 Carl von Clausewitz が指摘した「目的の階層性」[3]を用いたもので，軍事科学では「戦争計画の原理」とされている．

3 Myers の STS 分割[4]であり，ノードは最大抽象点と名付けられている．

4 JIS X 0121 「情報処理用流れ図・プログラム網図・システム資源図記号」8.2 処理記号，8.2.1 基本処理記号，(1)処理 には，長方形の記述箱について，「処理は，任意の種類の処理機能を表す．例えば，情報の値，形，意図を変えるような定義された演算もしくは演算群の実行，又は次に続く幾つかの流れの方向の一つを決定する演算もしくは演算群の実行を表す．」と規定している．これは修正した方が良い．

ⓒ 2011 Information Processing Society of Japan

Vol.2011-SE-171 No.22
2011/3/15

３．展開したプロセス/工程

図 3[6]は設計からテスト迄を含めた全体のプロセス/工程を示す所謂Ｖ型網である．上の図 3.a はハード,下の図 3.b はソフトの場合を示す．両者の左側は図 2 で検討した階層展開連鎖であり，右側はこれと逆向きの階層的統合の繰返しである．

両図の略中央，図 3.a では最上部に cable design，次いで Pattern design 等がある．左では，図面中のある点から他の点への論理的接続であった．この中間領域では，これらを物理的な位置情報間の接続に置換えるもので製造設計と云う．更に右でケーブルや印刷回路基板即ちモノになる．その右の統合点は試験/テストが行なわれる．初回には単に動作のみでなく，各種のマージンやタイミング余裕等をも調べられる．

下の図 3.b で略中間の所は一斉に Coding と書かれており，ここでソースコードに置換される．この右方では各種のソースコード群は纏められ，このレベルでテストされ，右上方に伸びて行く．

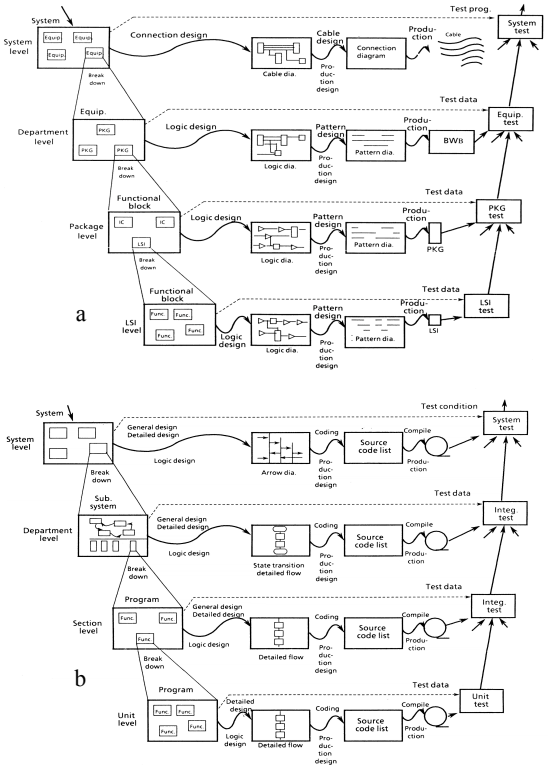


図 3　Ｖ型網

ハードであれ，ソフトであれ，初回のテスト対象の 1~数式分は，設計検証として厳密にキチンとテストされ,これらで設計の正しさが確認される．この範囲内ではハードとソフトには本質差は無い．もし差があるなら，ハードでは個々のハードの中に難しさが潜んで居り,技術的に難しい.ソフトでは実現手段を単純化したから，ハードのように難しさが塊状にある訳ではない．問題点は前記の階層展開連鎖が深いこと,上は経営レベルの問題から始まり最後にはコードになる間にある.従って，最上位は経営コンサルタント，次は事業所内の実務に詳しい方とこの種の分野に明るいエキスパート，以後はシステム作りだから自社内に沢山技術のある方がみつけられよう.発注者とこれらの方々とで決めながら進むのが一般的に当たり前であり，食違いが起らない．受託ソフト開発はこの常識論に立ち戻っては如何か？

この確認以後の生産過程はソフトとハードで大きく違う．ソフトでは確認済みマスターファイルから切出したファイルをメディアにコピーすればよい．ソフトの製造工程は最初から原理的に自動化されおり，その作業工数や製造コストは実際上問題ではない．しかし，ハードでは，図 3.a の右半分の production の部分から統合過程を経て右上の製品に至る過程は，製造台数だけ繰返し行なう（N 回グルグル回る）．この原理図ではごく簡単に見えるが，これは実際には巨大な階層展開と作業を経て階層的に統合されるネットワークで表される．ハードでは，１製品の設計を元にＮヶの製品が作られ販売される．従って製造の工数や部品等を含めた製造コストの負担の低減が大きな課題になり,各種の品質問題が大きくクローズアップされる．

この原理図では簡単でも実際には巨大なネットワークになり多量の生産が行なわれる．その中にはソフトウエアに良い参考になることも多い．幾つかをごく簡単に説明する．

１．打合結果は，席上で再確認はするが，以後は全てコミットメントになる．
２．科学的合理性：製品の規格である特性の上限値と下限値の中に，製品の製造時の実力特性値の上限と下限を納める．製品製造時のバラツキがσである時，　規格上限値から実力上限値および実力下限値から規格下限値の空きスペースの中に（例えば 3σ）が入るようにする．
３．約束には，必ず余裕（マージン）を見込む／余裕の無い約束は禁止されている．マージンは，平均的に起る諸トラブルの損失の期待値×発生率，とでも云える．マージンを食いつぶす事態が発生したら，まず自分が最大限の努力をして食止める．手に余るなら，早急に上長に報告せよ．上長は自分～自チームの他メンバも各自の予定を実行しながら応援して食止める努力をする．これが不足と感じるなら，課長更に部長と報告を上げ，組織的にトラブルを消し止める．組織の作業は必ず工程の階層と一致させ，日頃助け合う信頼が必要である．
４．製品の製造には，最高のヒトが当てられ，自動化や高級な機器類が充分に備えられ，改善や向上の教育は惜しみなく行なわれ，事業所随一の資本装備率で高い誇りを持つエリート職場であり，これらで一分のミスもスキもない高効率作業に従事している.「製造だから３K」とは全く反対の職場である．

ⓒ 2011 Information Processing Society of Japan

４．自由競争市場製品の特質

殆どのハード製品はメーカが自主的に仕様を決め，競争的市場で販売し，事業を運営する．（組込システムは上記の中の実現手段としてのソフトである.）各種のパッケージソフトや特殊な計算や各種の支援ソフト等も同じである．これらは自由競争的市場で勝たねば営業できない．各種の報道がなされているように，営業範囲が世界中に広がった結果，地球レベルの競走になり，世界的なトップレベル達が殆どの市場を押さえる寡占状態が進んでいる．実社会では前者が正常で，後者は非正常なビジネスとされる．前者は生死を掛けて闘っていることを理解されたい．2 章と 3 章でハードもソフトも展開過程は同一であることを説明した．自由競争市場製品と競走環境ではない製品とでは，同じ展開でもその影響の重要性は全く違う．充分にご理解頂きたい．

4.1 経営レベルの上流

競走的市場でのビジネスは「死ぬか生きるか」の厳しい環境であるから，企業の経営幹部は，自分達が自由競争市場で勝って成長する戦略を立てて実行する．

＊事業戦略の例を挙げる．元インドの鉄鋼メーカであったミッタル・スチールは 1989 年に創業した．各種の手段を通じて生長し，今や世界トップの鉄鋼メーカである．鉄鉱石の買入量は世界最大で，その買入価格も世界最低と推定される．M社に競争するメーカは M 社より高い鉄鉱石を買って鉄鋼材をつくらざるを得ず，しかし販売では，M 社が最大規模を活かした世界最低の安値で競争を仕掛けてくる．勝負に負けるか大赤字で受注するしかない．かくて競争社は順次滅ぼされる．このように，「戦略」とは発動すると必ず相手が負けに至る策を云う．

＊製品戦略について Apple computer のケースを説明する．この会社のパソコン Macintosh はパソコン市場の最古参の一つで独自 OS を特徴とする．しかし，これまでパソコン市場の僅かなシエアで生きてきた．2001 年にディジタル携帯音楽プレーヤーiPod を Macintosh 用に発売し，2002 年には Windows 用に拡張した．これはパソコンの iTune で音楽をダウンロードして蓄積し，再生する．2003 年には iTunes Music Store を傘下企業に開設し速やかに高い世界シエアを確保した．歴史的にはソニーはアナログ携帯音楽プレーヤーである Walkman を売ることで大きな利益をあげた．今回 Apple 社は再生機器と音楽の両者の販売で大きな利益を得ている．次は 2007 年に携帯電話 iPhone を発売した．これはデイジタル方式の携帯電話ではあるが，電話に閉じずむしろパソコンと同格に位置づけてあり，パソコンの下位(従ってより数量がより多い)の PDA (Personal Digital Attendant) を狙うと云われる．日本では携帯電話で各社が努力したが僅かなシエアに留まったが，世界市場向けの販売を自ら行なって，短期間に大きな世界シエアを獲得した．これは大型商品になる．2010 年には iPad を発売し，1 年間で 1500 万台売ったという．これは A4 版より小さいディスプレーのタブレット端末である．書籍を含む映像全般をカバーすると云われている．これらの新製品が世界レベルで急速に多量販売され，Apple 社の売上は停滞気味の Microsoft を脅かすに至っている．

＊開発戦略の例を説明する．1960 年代迄，精密な科学技術計算は歯車を使う手回し式計算機を使い，部屋中に騒音をまき散らしつつ何日間も掛けて計算した．これがリレー化次いで電子化されて，値段は小型自動車並みになった．やがてトランジスタ化され小型化し普及した．以後，集積回路・LSI 化されて個人毎のツールになり爆発的に普及し始めた．以下はこの時期の一齣である．新しい LSI を使うほどより高い集積度にできるから，トランジスタ等の個別部品が減って計算機は小型かつ安価になる．しかし，最低限買わねばならない LSI 個数は増えるから，より安値を掲げて売りまくろうと誰しも考えた．この拡大市場目掛けて多くのメーカが参入して一時期は 50 社が参入し，新製品をより安い価格で発売し，電卓戦争と呼ばれた．

C 社は初期のリレー時代から参入した先頭グループに位置していた．この経営幹部 K 氏は，全体の流れのパターンに気付く．即ち C 社等の先頭グループの新製品発表がひとしきり続くと，後発各社は先発各社の新製品群を調べ，これらに優る新製品を企画し，開発に着手する．しかし，先頭グループはより早く決断し開発して市場に製品を出すから，先発社から順に新製品群が発表される．このサイクル長は，製品の回路図を作り LSI のマスクパターンに展開する期間により決まる．

K 氏は従来通り，流れのパターン乗った「新 LSI 機」の開発を推進した．同時に，別部隊を編成して 1/2 サイクル遅らせて現在開発中の新機種を更に超えた「次期版 LSI 機」の開発も続けて走らせた．やがて「新 LSI 機」が完成し，他の先頭グループの新機種共々発売される．K 氏はそれを睨んで，各社が次の新製品を企画し設計して LSI 開発に突入し，最早引き返せないと云う時点で，C 社の「次期版 LSI 機」の発売を宣言し直ちに営業を開始した．他社は進行中の機種の開発を続行しても最早陳腐化したから開発費の回収すら見込めない．競走社は続々と撤退した．これおよびその他の何度かの「合戦」を経て，市場に生残ったのは C 社の他は，各種の技術で勝負を仕掛けた S 社のみであり，今は寡占市場になっている．

経営者には戦略を駆使して，競走市場で勝つ責任がある．同様に他の職掌にも，戦勝する責務がある．社長は中央研究所長に「，各事業毎の基幹技術の交代に併せて新基幹技術の準備や共通技術の展開」を命令する．営業部門の貢献は売上高で評価される．市場の様相，主要な競走会社の製品群とその販売状況および代表的なユーザの将来への期待等を社内に伝え，事業を先導しようと開発陣営に働き掛ける．製品企画/計画部署と開発チームのトップは，概況のみでなく中核ユーザや保守担当からの生の実態を把握し，開発部隊の実力と突合わせて内外を統合できる解を求め

ⓒ 2011 Information Processing Society of Japan

る．皆が生き残りを賭けて全力投球している．それは技術の合戦である．

4.3 製品企画から設計

ハード設計では「安い部品を数少なく使え」が徹底されており，一応出来上がった後にも“Value Engineering” または“Value Analysis”があり，部品をより安価な品に交代させる，より安い作業に設計改変をする等の見直しをして設計の原価意識の不足による不良は徹底的に叩く．ハードが「・安い部品を・数少なく使う」設計をすれば 1 台当たりコストが下がる．ソフトでは・小さな規模で・高い生産性で開発すれば，システム開発コストは下がり，・魅力ある機能/サービスを実現すれば，売上が増える．

ハード開発陣営では，本格作業前に下記を行なうことが常識である．自由競争市場向けのソフトならば，ハードと同様にせねばならない．

製品化以前に原理的/技術的/実行上の困難性等を評価して，以下の処置を決める．

・原理的レベルあるいは技術的問題があるなら，
　かなり前から研究機関に研究を依頼する
・軽度の技術的～実行上の困難性等があるなら，
　実行前に研究所/開発陣営がパイロットを先行させる
・行上の困難性なら，先行チームを作り作業方法を確立させる

開発作業中でも，以下を義務付ける．

・実行に当たっては，必ず複数の次元方法案を考え
　概略で良いから定量的評価を行なえ．事後に選択の正しさを評価せよ
・複数案があり，大きな負担の相違や結果が重大な岐路にあるなら，
　リーダやメンバに必ず図って検討せよ．

ではソフト設計の現場では，これらがおこなわれているであろうか？受託開発を行なうベンダーでは行なっていないし，行なうことは困難である．以下に実態分析例を説明する．図 4 [7]は略同様な機能/サービスを持つ或る種のシステムの中心部分のソフト規模であり，3 実績(X, Y および Z)の規模を正規化して示した．夫々の特徴的な取組みを以下に示す．

・Z は依頼に応えて開発する（規模削減等が全く脳裏に無い）ベンダーが，一応の教育を受けた後に開発したシステム，

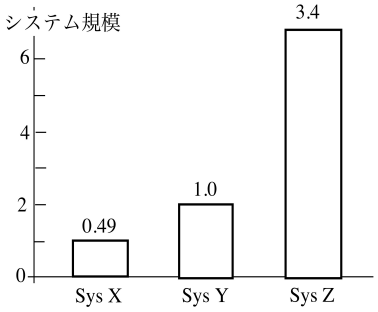


図 4 実例 3 種

・Y はハードの布線論理で作られた同種システムの設計者らが，ベンダーのソフト設計者の作業結果を見て規模を減らす工夫や修正を共同作業したシステム，
・X はソフト規模削減の研究結果を織込んだシステムである．

３実績の各母胎の全体の大きさの比は殆ど上記３実績の比に等しい．従って，３実績例の規模の比は，３実績の「特徴的な取組みの違いから起った」と考えられる．

更に詳細を分析した．図 5[7]の B は前記の System Z（図の軸の右でシステム B と記した） と System Y（図の軸の左でシステム A と記した）の運用機能を取上げた．基礎的な機能部分の規模は（運用機能を指定するコマンドの細分化指定）に比例することがわかったので，これを単位に表わし規模を評価した．

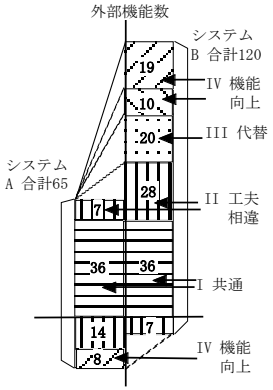


図５　2 実績例の運用

同じ基礎的機能の範囲で総数は（A 65 : B 120）と B は約２倍大きい．軸の左右の帯状領域のハッチは両者の対応する同一細分化機能（規模）部分を示す．両者の内，基礎部分を中心に横線ハッチの 36 は共通である．A のその上縦棒ハッチの８は，B では 28 にふくれている．軸の B の右上の・ハッチの部分は，A では他の代替手段で用を足している．横基線の下の縦棒ハッチの部分を見ると，右では 7 で済ませている部分が左では 14 を要している．これは A の技術不足である．これらから，左の方 A が各種の工夫をして削減している様子が伺える．次に図の左の最下部の 8 は左側 A の付加分で，右の B では最上部で（ランクの差があるが）気前良く 29 も奢っている．これらから左の A では極力減らそうと工夫し，努力していることが明白である．

最初の図４に戻り，System X の規模が Y の規模の約 1/2 である理由を説明する．この図の部分は所謂組込みシステムで，有限状態機械（Finite State Machine, FSM)でモデル化できる状態を持つシステムである．オートマトン理論によれば，相互に独立な下位の m 状態の FSM_m と n 状態の FSM_n とで上位の FSM の機能を果たすようにする．FSM の状態数は最大で m×n になる．これを直接にプログラム化したものと，下位の 2FSM を組合わせた場合とのプログラム規模の比は　(m×n) : (m＋n)　になる．今 m = n = 10 ならこの比は　100 : 20 = 5 倍も大きくなる．Sys X はこの FSM の分割を行なった．FSM の分割は「知る人ぞ知る」技術であるから，実地試行等を経て行なったと推定した．結論として，Sys X は研究を行なったから，規模が削減できた．これは事前の調査や研究が大きな効果を示すもので，常識の通りである．

ⓒ 2011 Information Processing Society of Japan

斯様な結果が生じるには，それなりの原因がある．上記のケースおよびその他の場合に，実地に設計している被派遣者や請負担当者の訴えは以下の通りである．

「考えて設計してはならない．なぜなら枠組みを考えて最適構成にしても，発注側の進め方が変ると全滅してしまう.」

「云われたように，ドンドン作業しないと困る．全体はなかなか進まないものだから，進められる時に，ドンドン作業しなさい.」

彼等には彼等の立場がある．

これより上位の研究については表１の実績がある．これは日本の代表的なシステムインテグレータやソフトウエアベンダー等約 700 社の業界団体である日本情報サービス産業協会（JISA）の統計[8]で，1994 年から 2010 年の会員企業の平均 R&D 比率である．長期にわたり約 1%程度である．1%程度の低率の産業は鉄鋼業などの品種当たりの売上の極めて高いものである．平均的な産業では３％以上であり，先進的な企業では 5~7%程度である．

表１　ソフトウエアベンダーの平均研究開発比率

年	’94	’95	’96	’97	’98	’99	’00	’01	’02	’03	’04	’05	’06	’07	’08	’09	’10
R&D 率(%)	1.10	1.04	1.33	1.58	1.62	1.72	1.2	1.1	1.0	1.0	1.1	1.02	1.15	0.84	0.47	0.82	0.81

研究開発費率の国際的な値を探すと，かような国レベルでの統計は見当たらない．しかし，優秀企業は自社の値を公開している．ソフトパッケージ系メーカーは 10%に近い値から 10%以上である．往時の Microsoft で Bill Gates は株の配当を行いたがらなかった．それは競走社が追付くことを恐れ，全ての余力を競争者を引き離す為の研究費などに回したかった為と云われる．

受託開発を中心とするソフト企業ではオフショアが叫ばれ，次々と労賃の安い国が話題になる．若者達は,「これは労働集約産業」との認識が高まり，この分野は優れた人材が志望しなくなっていると聞く．

実態分析を第一線から経営トップの指標迄説明した．この世界は自由競争的市場で必死に生きようとするカルチャーとは，全く異質である．適当に

作業してもらう必要がありえるから，作業依頼法を充分に準備する他ない．また，学会等の技術の場も現状とは全く別建てに準備すべきである．

５．設計の管理

５．１　プロセス

設計と製造の両プロセスを並べて下記の対比を考える．

製造は作業内容が事前に　確定済．設計は作業内容が事前には未確定．

製造のプロセスの管理は　確立済．設計のプロセスの管理は　確立済でない．

設計管理の難しさは上記の未確定/未確立から生じる．本質的には両者は同じ基本構造だから，製造のプロセスの管理を設計にも流用できる．特に製造作業の対象は具体的なモノであるが，管理の立場からは前記のモノから得た情報を扱う．故に管理の仕掛けも共通になりえる可能性がある．本章ではこれら流用を中心に紹介する．下記は製造側からみた工程に関する基本条件であって，ハード設計は殆ど下記を取込んでいる．各種のプロセス/工程の中でも，責任と権限を持って直接に作業する担当者が構成するプロセス/工程が，QCD に最も深く関係する．従ってハードの設計や製造および自由競争市場で営業するソフトでは，ここに重点を置く．

所謂ソフトでは実態中心ではない[5]ので，[]でコメントを記入した．

１．工程（プロセス）は階層的構成を基本とする．[ソフトでは反対者が居る]

２．工程は，入力から出力への広義の変換であり，入力と出力をノードとして着目する．即ち，端面で規定し，内容では規定しない．[IE では内容を定義すると自己撞着に陥るから端面主義を採る．所謂ソフトでは未だに渾然]
工程名は内容を反映するが，定義ではない．

３．工程と職場の作業とは（工程の両端面の定義に従い）階層構成も合致させる．

４．工程の管理は（共通的な工程管理業務で工程員が行なうから），工程内の処理に直接には関わらず，それは別途の技術として処理する．[工程管理が未分化なので，それほど画然としていない．]

５．工程の特性は基本的に外部で計測できる値から得た「外部特性値」を使い，基本的に内部には立入らない．[同上]

６．内部を知るには，原工程を階層展開した下位工程の外部特性による．[同上]

７．工程を厳しく統制することは，工程の定義（例えば誤差や仕上げ精度等）を厳しくする（誤差等を小さくする）ことである．手引きとして工程内部の作業順序等を記した概要図を使うが，これは定義ではない．[同上]

５　CMU SEI は米国防省から政府が発注するソフトの発注条件の研究を行なった．その結果，現規定のように中核部分に立入ことを避けてアセスメントに留め，周囲環境を厳しく統制する方式が産まれ，受注者の具備条件になった．受託開発企業にとっては資格の有無は重要である．これらから手引書には実態ソフト工程の改善は放擲して形式条件の整備中心のものすら流布している．

ⓒ 2011 Information Processing Society of Japan

5.2 ソフト作業の外部特性値は大きくバラツク

ハードは，過去 100 年以上を掛けて，product と process の改善を行なってきた．その結果，殆どの外部特性値は正規分布状になった．しかし，ソフトは過去の蓄積も充分でなく市場部品のような外部の応援もないから，ソフトの各種の特性値は皆バラツキが大きく（平均値の約 1/3 倍から平均値の約 3 倍の間に渡る）対数正規分布も珍しくない．それはソフトが他とは異なる訳ではない．ソフトはこれから努力を蓄積して改善せねばならない．図 6 はその平均値とバラツキの様相を示す．統計学に従えば，平均値の世界とバラツキの世界に分けられる．特性値の改善策を行なえば，図の平均値のカーブのように向上して行く．これは振幅を小さくするから，同時にバラツキも小さくなる．

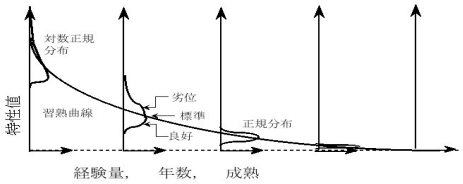


図 6　特性値の向上とバラツキの低減

5.3 定量計測し評価することで「分割し統治する」

図 7 [2]のように，まず工程に区分する．図の Design では下位の４工程毎の特性値を計測し評価する監視を続ける．夫々定義も違い値も違うが（定量評価はシャープであるから）４工程の内どれかの異常あるいは相対劣位が検出し易い．Process design が問題なら，その工程の中間に２図面を入れて順次に図面を書かせチェックさせれば，右図の弦の振動様の図に示すように，中間が押さえられ，従来の 1/2 サイクル状から 3/2 サイクル状に統制されて，(3 分割なら)1/3 に低下する．（品質不良の場合には細分化工程での設計直後の机上チェックを強化する．残留率は 1/3 に低下する．）未だ不十分なら更に階層的に下位に展開し細分化を繰返す．

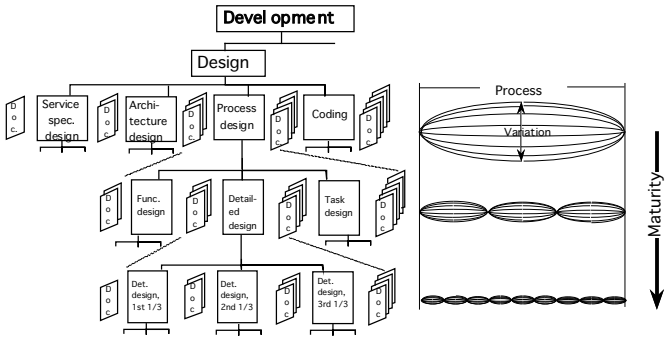


図 7　分割し統治する原理図

このように，まず工程のバラツキを減らす．これは生産性と品質の両方に効き，仕事が安定化する．工程/プロセスは，入力を処理して出力にする．各種の特性値にもっとも影響するのは処理である．この処理とは，過去に頭脳に蓄積した記憶を引き出し，適当にアレンジして出力欄にペーストする操作である．

改善し向上させる対象は，まず第一に貴方自身で，周囲環境ではない．次には前位/対面する顧客等からの影響であり，その次は貴方を評価し指導する上長である．所謂ソフトウエアプロセス等はこれらに比べ影響はごく小さい．品質向上や欠陥率低下を実現するには，貴方自身が誤りの無い人間に変わらねばならない．生産性向上を云うなら，貴方の頭脳が優れたものに変らねばならない．高い成熟度のチームの人と話してみられると良い．ものごとを受止めて正確に話され，技術に明るく，そして礼儀正しく，明るく前向きに受止める．これらが揃うから創造中心の高いレベルの仕事ができる．問題はプログラミング等ではなく人間であり，新技術の吸収や自己あるいは他の作業分析から改善とその評価など，学ぶべきことは沢山ある．

5.4 改善する努力を繰返し続ける

人は経験を重ね，経験して得た知識を蓄積し再利用すると，製品 product と工程 process を改善できる．これは習熟効果と云われる．図 8 はある開発チームのプログラム開発の作業時間を正規化した生産性で示した．図 a[9]は直線尺度表示で，経験

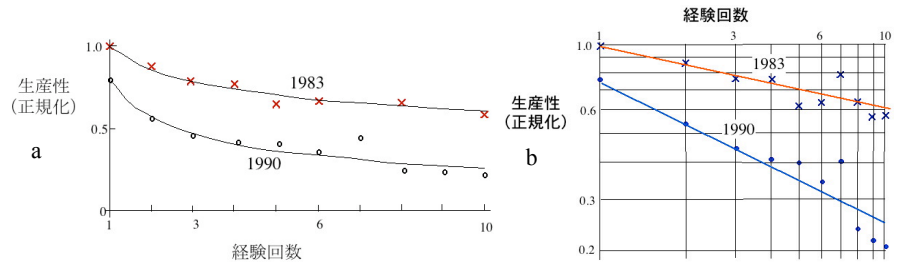


図 8　対数習熟効果

を重ねるにつれて，始めは急激に向上するが，その度合いは次第に低下する．図 b[10]

ⓒ 2011 Information Processing Society of Japan

は横軸の累積経験量と縦軸の生産性の両対数尺度表示で，直線状の傾向線を示す．これは対数習熟効果と云う．人の殆どの特性は対数習熟効果を示す．
・1983, 1990 年の曲線はチームメンバーの脳裏に知識が蓄積され生産性が向上．
・左縦軸 1983 年から 1990 年にプログラム自動生成機能が追加され生産性が向上．
前者は意識せずに知識が獲得され向上し，後者は意図して設備を投入したから向上した．向上努力/新知識が増すにつれ，生産性があがる．努力当たり改善が大きければ両対数尺度図での勾配はより大きくなる．QC 運動等でチーム毎成績をプロットした図を見ると向上や停滞が一目で見える．

6. おわりに

2 章および 3 章は，ハードとソフト，および設計と製造には，本質的な知的処理は全て同じことを説明した．従ってソフトがハードその他から違ったものではありえない．一方，実行の様態に於いてソフトの製造は原理的に初めから自動化されており，一方ハードの製造は原理的に各種の改善向上を必要とした．100 年間以上にわたるハード製造の管理技術と作業の機械化等の向上努力の蓄積の結果，現在では製造部署は各企業とも最も大きな設備投資がされる先進部署で管理もずばぬけたレベルにある事実をご認識頂きたい．

実際上の大きな差異はソフト/ハードではなくて，営業する市場が下記の
・世界トップでなければ生き残れない，厳しい自由競争市場
・極めて相互競走が少ない特殊な市場
いずれであるか，から生じる．4 章はここに焦点を置いて説明した．両者は基本環境が違うから，相互に相容れることはなさそうと思える位，違いは大きい．本学会では後者のみスポットライトが当たって，前者は弾かれる立場である．逆転することが望ましいのではないか？

ハード製造現場の中には設計にも活用できる各種技術が多数ある．5 章は幾つかを紹介した．

終に再度問題提起したい．厳しい自由競争市場で営業する各企業に役立つ技術を向上させ，彼等に役立てること，それが工学の第一の責務ではなかろうか．

謝辞

著者等は，アカデミーと実社会の両方を経験し，学ぶ機会に恵まれた．長年の間に色々な方々から頂いたご指導に感謝します．

参考文献

1 河野善彌，陳慧，アボールハッサン，H.：人の設計知識構造と定量評価（1/2 ～ 2/2), 信学技報， KBSE2003-57~58, pp. 67-78, 2004.
2 Koono, Z., Chen, H. and Abolhassani, H.: An Introduction to the Quantitative, Rational and Scientific Processes of Software Development (Part 1 and 2), SOMET 2007, pp. 361-390, 2007. 著者邦訳，URL http://www.creationproj.org
3 Clausewitz, Carl von, Vom Kriege. 訳本：淡徳三郎，戦争論，徳間書店，1965.
4 Myers, G. J.: Reliable Software through Composite Design, Petrocelli/Charter, 1975. 訳本：国友義久，伊藤武夫，ソフトウエアの複合／構造化設計，近代科学社，1979.
5 Jackson, M. A.: Principles of Program design, Academic Press, 1975. 訳本：鳥居宏次，構造化プログラム設計の原理，日本コンピュータ協会，昭和 55 年.
6 Koono, Z. and Soga, M.: Structural Way of Thinking as Applied to Quality Assurance Management, IEEE Journal on Selected Areas in Communications, Vol. 8, No. 2, pp. 291-300, Feb. 1990.
7 Koono, Z., Kondo, T., Igari, M. and Ohtsu, K.: Structural Way of Thinking as Applied to Good Design (Part 1. Software size), IEEE Communications Society " IEEE Global Telecommunications Conference 1991", pp. 24.3.1-8, 1991.
8 日本情報サービス産業協会，JISA 基本統計調査，毎年． URL http://www.jisa.or.jp/statistics/index.html.
9 森岡洋介，長野文子，大野治：EAGLE/P を用いたプエオグラム開発の習熟，情処全国大会（平成 3 年後期），1K-3, 1991.
10 河野善彌，陳慧，アボールハッサン，H.：人の意図的行動の知の構造とそのはたらき，信学技報 KBSE2006-56, pp. 7-12, 2007.

ⓒ 2011 Information Processing Society of Japan

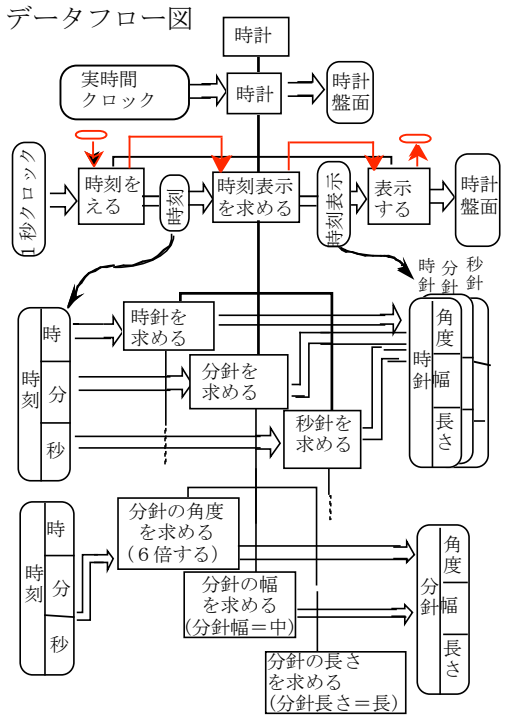
箇所	誤	正
2 頁　左	図 2 脱落	

図 2. 拡張構造化設計　「時計」

ⓒ2011 Information Processing Society of Japan