第4回

日本の組込みシステム開発の特徴と今後の展開

奥村　洋　（株）ガイア・システム・ソリューション
yo@gaiaweb.co.jp

ものづくりと組込みソフトウェア

バブル崩壊以降10年以上続く経済不況の中にあって，日本の貿易収支の黒字を支えているのは製造業の国際競争力である．製造業の生み出す工業製品の中に含まれる組込みソフトウェアは製品の複雑化に伴って増加し，現在では開発コストの3割を超えている．日本の競争力という視点から，組込みソフトウェア技術の重要性がクローズアップされている．

■アーキテクチャの比較優位

現在の日本の製造業の強み弱みについて明確な視点を与えているのが，藤本[1)]によるアーキテクチャの比較優位論である．バブル以降，90年代の劇的な変化の中で，日本の製造業は少なくない分野で競争力を失ったといわれている．しかし，ブランド力・財務体質・販売チャネルなどの表の競争力の弱さに隠れてはいたが，裏の競争力であるものづくり力は依然健在であり，それが今の日本を支えている．ドラスティックな価格競争に巻き込まれなかった領域では，技術水準が高く品質に対する信頼性がある日本製品の価値が高く，依然として強みを持っている．

アーキテクチャの比較優位では，ある国がある工業製品で競争力を持つのは，その国特有の組織能力が製品アーキテクチャと相性が良いからである，と考える．アーキテクチャを，インテグラル／モジュラー，オープン／クローズドに類型化すると，日本が急速に競争力を失った製品はオープン・モジュラー，引き続き日本が競争力を保っているのはクローズド・インテグラルアーキテクチャの製品であって，それは日本の製造業の持つ統合型ものづくりの組織能力とクローズド・インテグラルアーキテクチャの相性が良かったからである，と分析されている（**図-1**）．このことは現場の感覚や非競争領域の製品が強いという事実とも合致している．

ここでのアーキテクチャの定義は，「1つの製品を構成要素や工程に分割し，機能を分配してできる，要素間・工程間のインタフェース上の決まりごと」である．インテグラルアーキテクチャとは，複数の設計パラメータが強い相互依存を持っており，統合機能を提供するには，要素間の擦り合せが必要な設計構造のことで，モジュラーアーキテクチャとは，設計要素間の依存関係がないか，あってもごく弱く，統合時にあらかじめ設計された機能性が出るような設計構造のことである．

■組込みソフトウェアとアーキテクチャ

今日さまざまな工業製品に組込みソフトウェアが含まれているが，もともと機能をソフトウェアで実現することの効果は，製造コストの低減を目的としたものであっ

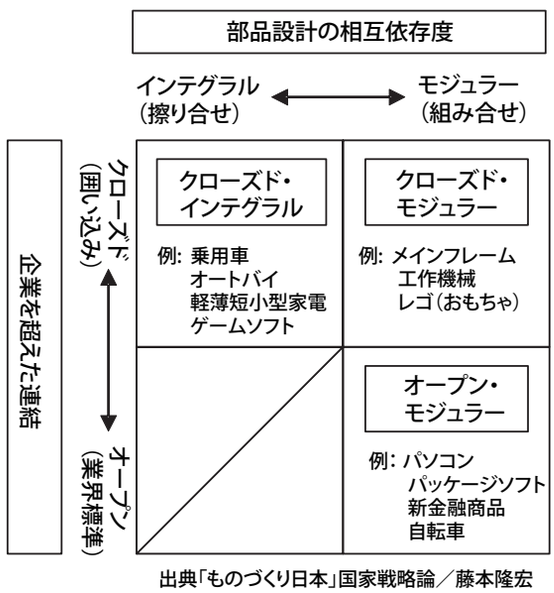

図-1　アーキテクチャ類型

た．システム統合と機能提供をソフトウェアで実現することで，機械部品や電子部品の点数を削除できるのが大きなメリットであった．つまり，製品アーキテクチャの中でも組込みソフトウェアは重要なつなぎ役の部分を果たしていたといえる．製品の機能が増加するにつれ，ソフトウェア量が増大したため，組込みソフトウェアは製品アーキテクチャの側面からも重要な技術領域になりつつある．

日本の国際競争力の源泉が工業製品で，工業製品の競争力の一翼を担うのが組込みソフトウェアであるなら，組込みソフトウェアの競争力強化への取り組みもまた，日本の工業製品の背後にある競争力に学ばなければならない．

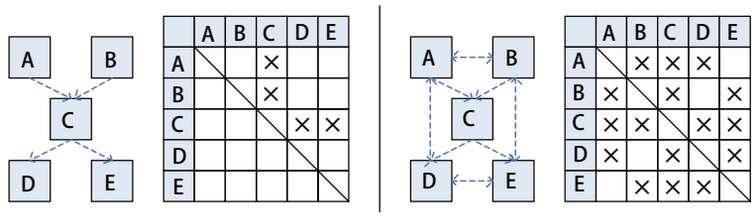

図 -2　依存関係のマトリックス表現

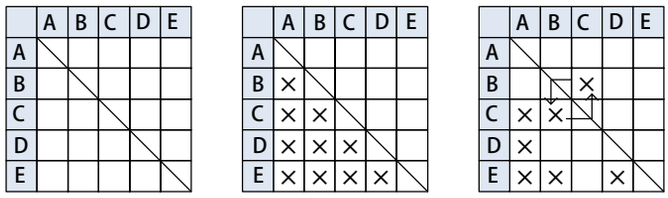

図 -3　タスク構造マトリックス

組み合せと擦り合せ

組込みソフトウェア開発が，今日クローズアップされている背景には，システムの高度化による大規模化・複雑化がある．英語で複雑を意味する“com-plex”は“consisting of inter-connected or interwoven parts”，訳すと「システムの構成要素が依存関係で接続されていること」であり，つまりはアーキテクチャの複雑さのことを意味している．

構造と複雑さ

設計要素間の依存関係はマトリックスで表現することができる（**図 -2**）．直感的に左より右の構造が複雑なのは，マトリックス中の×印の密度が高いからである．N 個の要素からなるマトリックスの場合，潜在的には N × (N-1) の依存関係が発生する可能性があるため，要素数の増加に対して潜在的な複雑さはその 2 乗で増加する．

組込みソフトウェア開発の複雑化の理由には，

1. 携帯電話などのように，ディジタル製品の高機能化・小型化のため部品点数を減らし，単一マイコンでソフトウェアを動作させるために統合が発生したケース
2. オフィス複合機のように，コピー・ファクス・プリンタのような要素技術に重なりを持つような製品が分割され再統合されるようなケース
3. 自動車の電子装備のように，ネットワーク化により個別機能が接続され，統合機能が提供できるようなインフラが整ったケース

などがある．

いずれのケースも，既存の機能部品が接続された結果，個別部品開発に比べて潜在的複雑度が 2 乗的に大きくなっている．

設計構造とタスク構造

設計構造上，A←B のように，モジュール B がモジュール A に依存している場合，

1. B の設計作業は A の仕様が決定しないと開始できない
2. モジュール B はモジュール A が完成しないと統合動作ができない
3. モジュール A はモジュール B が仕様変更されたら仕様を見直さなければならない

といった依存関係が発生する．このように，設計構造とタスク構造は基本的に同型になる[2)]．

図 -3 はタスク間の依存関係を図式化したものである．X は各作業がまったく独立に並行して実行可能であること，Y は各作業を逐次実行（ウォーターフォール）できること，Z は後工程からのフィードバックを必要としており，手戻りが発生する可能性があることを示している．

Z の場合，手戻りを減らすためにタスクの順番を入れ替え，極力左下に依存関係が集まるように工夫する必要がある．入れ替えの結果すべて左下に収まれば逐次開発が可能になるが，現実問題は並べ替え後も対角線右上に依存関係が残ることが多い．このような後工程からのフィードバックを必要とするタスク構造の解決戦略は基本的に 2 つある．

1. タスクをまとめて実行し，同一工程で処理する
2. 前工程は結果を仮決めして後工程に渡し，後工程からフィードバックを受ける

依存関係が対角線に近い場合は 1. で対応できるが，対

角線から離れている場合は2.で対応するしかない．1.を実行することは工程をブロック化することであり，全体工程を大括りに分割し直すことと同義である．工程を分割し直した結果，ブロック単位の依存関係が左下に収まればYに，対角線だけに収まればXにすることができる．

規模が大きくなると，これらの大括りにされたタスク単位で異なった組織に割り当てられる．異なった組織間で2.が発生するとフィードバックループが大きくなるため，工程上は望ましくない．このような工程は大量生産の製造工程ではそもそも許容されないが，設計工程では品質確保のために微修正からやり直しまで程度の差はあれ，フィードバックが発生することが多いため，ある程度許容されている．特に組込みソフトウェアは設計工程にかかわるため，品質確保のための微修正は不可避的に発生してしまう．

設計工程では，期間短縮のためにYからXへの移行の工夫が必要になる．要求→仕様→設計→開発→テストという開発工程は並行して実施することができないため，並行開発をするためには，モジュール単位に分割せざるを得ないからである．

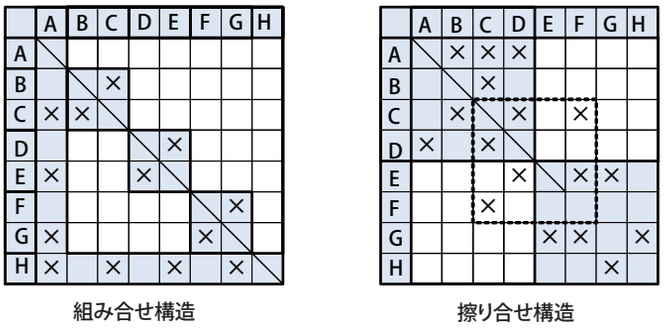

図-4　組み合せ構造と擦り合せ構造

■組み合せ・擦り合せ

並行開発を可能にするためには，ブロックの依存関係が擬似的にXの形になるように分割すればよい．この分割戦略の典型が組み合せと擦り合せである（**図-4**）．

組み合せの戦略では，Xの形にするべく，依存関係を周辺に押しやる戦略を採る．対角要素以外に依存関係がある場合は，その列をなるべく左に寄せる．特に他の多くの要素と依存関係を持つものを左に寄せると効果が高い．組み合せ構造では，L字型の部分を除けばXの構造になっていることが分かる．L字の最左列はすべてのモジュール開発を並行して実施するにあたって決めておかなければならない「設計ルール」を表しており，最下行はすべてのモジュール開発が終了しないと実施できない「統合」を表している．

擦り合せの戦略では，モジュールに分割した後，ある程度の依存関係を対角に押し込めて，残った依存関係は許容する戦略を採る．許容された依存関係は開発中も統合時もモジュール間の擦り合せで処理される．

組み合せ戦略を採ったとしても，L字部分が厚くなるとモジュールを並行開発するメリットが出にくくなる．そのため設計構造の全体を理解した上で注意深く設計ルールを定義しなければならない．この作業には経験・知識・能力と全体を見渡した構想力が必要で，アーキテクトと呼ばれる特別なエンジニアが必要になる．

擦り合せの戦略を採った場合は，全体システムを実現するために必要に応じて擦り合せを実施すればよいため，統合製品としてのシステムの品質は上げやすい．プロトタイプ，1次試作・2次試作といったかたちで統合機会を増やし，擦り合せながら依存部の確定をしていくことができるからである．しかし，開発を実施する場合は，モジュールをまったく独立して開発することはできないため，仕様調整や変更に伴うコミュニケーションが都度発生してしまう．完全に仕様を固めて完成品を納品してもらうスタイルの発注はしづらく，既存モジュールに手を加えないと統合できないといったことも起こり得る．

一般に，システムの統合品質を出そうとすると擦り合せが必要になることが多く，専用部品と専用アーキテクチャを利用した製品のほうが統合品質を上げやすい．

アーキテクチャと組織能力

日本がインテグラルアーキテクチャの製品を得意とするのは，その組織能力が擦り合せ型の開発に適しているからである．組込みソフトウェアの開発でも同様の傾向を見ることができる．

■組織能力の働き方

高度成長期の日本の組織は，和をもって由とする集団的メンタリティ，濃密なコミュニケーション，あうんの呼吸などの特徴を持っていた．そのベースにあるのは米国的な個人主義より関係性を重んじる東洋的な思想，上下関係を重んじる儒教的思想，農耕民族的思想であるといわれている[4]．これらの認識はいかにもステレオタイプであるし，今日の日本では変わりつつあるが，認知科学的説明は直感的に理解しやす

く重要な示唆を与えてくれる．擦り合せが得意な風土はTQC/TQM[☆1]を生み出した風土であり，セル生産方式を得意とする風土でもある．

ものづくりにおいては，現場主導の改善が最も重視される．現場のリーダは優秀なエンジニアであることが多く，一子相伝的にエンジニアを育成し技術を伝えるという職人文化の中で育ってきた．同時に現場の改善は現場の人間が最もよく分かるのだという前提で，現場発の改善活動が奨励された．まずやってみて，改善点を発見し，施策を実施し，その結果を吟味し施策を考える，というサイクルをまわす，という改善マインドの浸透をもって組織能力をエンジニアリング力に結び付けてきた．

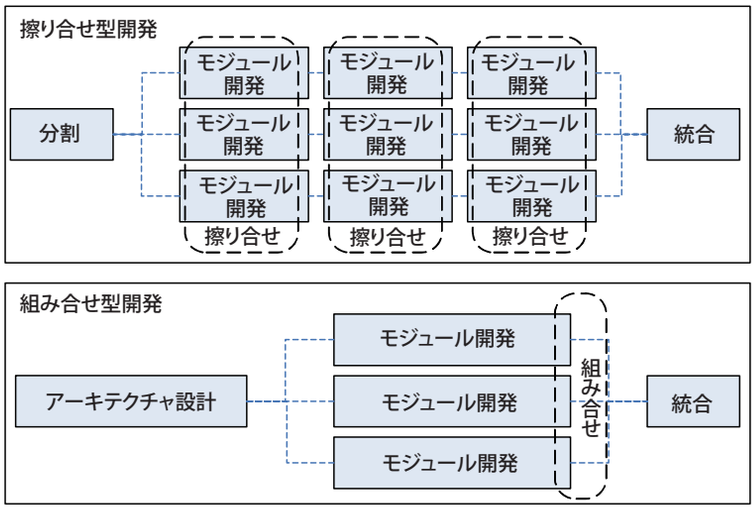

図-5　擦り合せ型開発と組み合せ型開発

こういった組織能力の働き方が擦り合せ型のシステムに適しているのは，分割したモジュール間での調整を統合までに必要に応じて繰り返す擦り合せ型の開発に対応できるからである（**図-5**）．

創発的な組織の働きの違い

90年代の米国では，いわゆるモジュール化現象によって，イノベーションの推進主体が大企業中枢からモジュールごとの非集権的で局所的な小集団に移動し，シリコンバレーに代表されるベンチャー企業を主体としたイノベーションが生まれたという[2]．逆に日本は，80年代の製造業の成功体験から抜け出せず，オープン・モジュラーにシフトすることができずに，少なくない分野で競争力を失ってしまった．しかし，それが直ちに日本のイノベーション能力の低さを意味するわけではない．組織能力が生み出す創発が発生するコンテキストが異なるだけである，という理解が妥当であろう．

80年代の日本の人事制度や雇用制度などの社会環境は，創造的活動を制限するという欠点がある反面，短期的に成果が出しづらい活動も低いリスクで実施でき，結果として現場発の創意工夫に貢献していた．オープン・モジュラーが企業を超えてモジュールクラスタを形成し，創発的に価値を生み出すのに対し，クローズド・インテグラルでは，企業，事業部，系列など，あらかじめ規定された枠の中で創発的活動が起こり，その枠の中で品質を高めるかたちで価値が生み出される（**図-6**）．

かつてまがいものの代名詞であったメイド・イン・ジャパンがいつしか低価格高品質を意味するようになったことは，日本がもともと革新的な製品を生み出すことよりも，すでにある製品をより早く安く高品質に作り出すことに対して能力を発揮するということを示している．擦り合せ型の開発は作るものが決まっていて統合品質を上げていく場合に適していて，組み合せ型の開発が新しい革新的な製品を生み出す場合に適しているとすれば，意図的かどうかは別として高度成長期の日本は総じて擦り合せ型の戦略をとってきたといえるだろう．

製品の高品質化を目指す品質工学が日本から生まれ，ビジネスストラクチャを視野に入れたモジュール化理論が米国で発達したことは示唆的である．

組織・構造・プロセス

このような組織能力の働き方はソフトウェアプロセス改善の現況にも表れている．今日のソフトウェアプロセス改善の代表格はCMMI[☆2]であり，各社こぞってCMMIの導入によるソフトウェア品質の向上を目指している．しかし，CMMIの導入で成功を収めた日本企業はさほど多くない．部門内の局所的な成功にとどまることが多く，中には完全撤退を決めた，という企業もあり，各社試行錯誤中であるようだ．

誤解を恐れずにいうと，導入失敗の原因は，CMMI的なアプローチは組み合せ型システム開発の文化のもとで発達したものであって，擦り合せ型システム開発の文化

☆1 TQC：Total Quality Control: 全社的品質管理，TQM：Total Quality Management: 総合的品質管理
☆2 Capability Maturity Model Integration：能力成熟度モデル統合

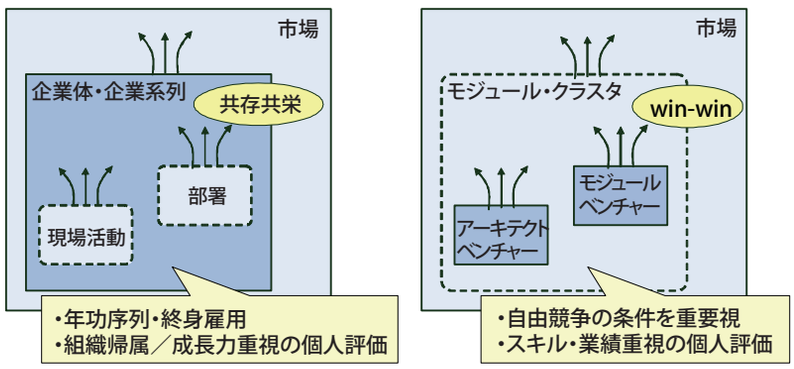

図-6　創発的な組織力の働き方の比較

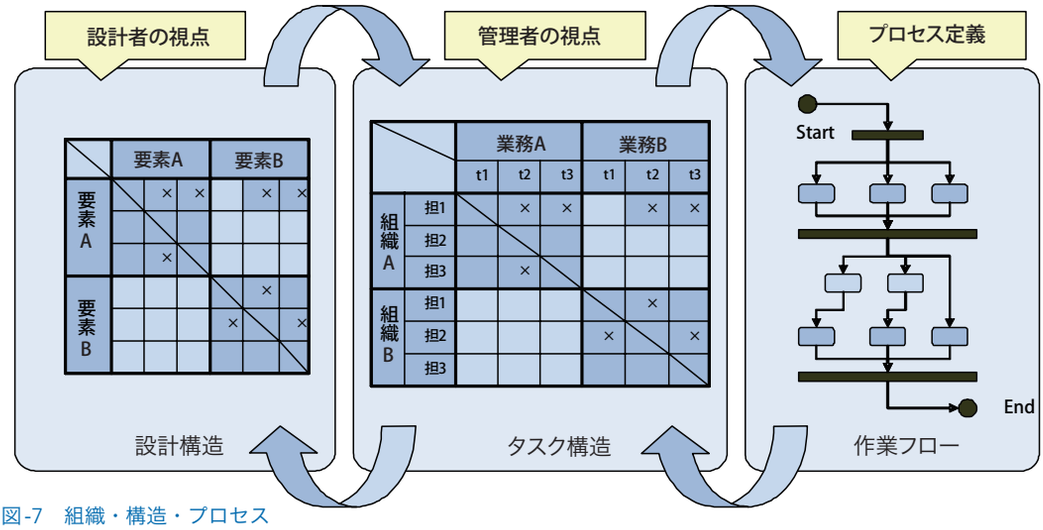

図-7　組織・構造・プロセス

とは馴染みづらいところがあるからだ[3]．プロセスはタスク構造を通じて設計構造と結びついており，設計構造と関係なくプロセスが決められるわけではない（**図-7**）．

擦り合せ型の開発では，ある程度の依存関係を残したまま繰り返し統合しながら開発を進めていく．その過程はアーキテクチャの洗練過程でもあり，プロセスの洗練過程でもある．設計構造とプロセスは密接に関係しているため，アーキテクチャがほぼ固まった後でなければプロセスが確定しない．

基本的にCMMIなどのプロセス改善活動自体は単独では何も生産しない．いかにCMMIのレベルを上げようとも，それはプロセス改善力が増したことを意味するだけで，改善活動自体は現在の活動に対してはコストアップになってしまう．

ソフトウェア品質改善活動で槍玉にあがる手戻りや仕様変更は，単純に悪と考えられがちであるが，擦り合せ型開発の場合は別の解釈も必要である．すでにみたようにフィードバックループは当たり前のことであるため，手戻りや仕様変更は品質の作りこみ作業であり，ムダな作業というより価値生産活動であるという解釈である．問題は途中段階の品質計測の手法が確立されていないために，無計画な手戻りやムダな手戻りがあったり，手戻りの改善が進められなかったりすることであって，どんな手戻りもすべてなくしてしまうことを目標とするのは無理がある．

このような特性を無視したためにスムーズな導入に失敗している例はPMBOK[☆3]，UML[☆4]の導入などにも見られる．

一方で，業務系・情報系システムのインテグレータを中心に，CMMIやPMBOKが着実に浸透し，成果を挙げている．この違いはホスト系からPC系への移行が済んだこと，構築プラットフォームが急速に整備され，アーキテクチャが組み合せにシフトしたため，組み合せ型開発がしやすくなり適用効果が出やすくなったことに成功の理由があるようだ．本来の擦り合せ力は，弱体化することなく，システム開発から要件開発に振り向けられる

☆3 the Project Management Body Of Knowledge：プロジェクトマネージメント知識体系
☆4 Unified Modeling Language

ようになり，統合品質を上げる方向に働いている．

■擦り合せ的な組込みソフトウェア

組込みソフトウェアは当初はハードウェアのおまけでしかなく，組込みソフトウェアエンジニアという種類の技術者は存在しなかった．ソフトウェアの開発規模が増大するにつれ分業化がなされ，ハードウェアの知識を持ったソフトウェアエンジニアに暗黙知として引き継がれ，職人文化ともいうべきカルチャーの中で各社各様に組込みソフトウェア技術者が育成されてきた．時間制約がある組込みソフトウェアのアーキテクチャは擦り合せ型であり，その設計工程をエンジニアリングで捉えることは困難であったが，擦り合せ型の組織能力によって結果として製品の高品質化を実現してきたといえる．

統合品質の向上と組込みソフトウェア

これまで，組込みソフトウェアの世界では統合品質を上げるために擦り合せ型の開発が功を奏してきたように見える．しかし，今日の複雑度の増大に職人集団的アプローチで対応するのは非常に困難になりつつある．統合品質の出しやすさを盾に擦り合せ一本槍で突き進むと，モジュール化に対応できずに競争力を失った産業の二の舞になりかねない．

■組み合せと擦り合せの遷移

組織や開発方法の組み合せ・擦り合せの選択は設計構造に依存し，設計構造の組み合せ・擦り合せの選択は，ユーザが要求する統合品質に依存する．つまりビジネス的視点で選択されるべきものである．

ある新しいカテゴリの製品が登場するとき，当初は擦り合せ構造であることが多い．ユーザニーズや要求品質がまだ安定していないため，機能性の改善が頻繁に発生し，高度なモジュールの擦り合せが必要になるからだ．

製品進化がある程度進み，ユーザが要求する統合品質が明確になると，擦り合せは過剰品質を生み競争力の低下を招く．そのとき，同じ製品で低価格化を目指すか，より高品質なニッチ市場にシフトするかの選択を迫られる．

低価格化を目指すには，擦り合せ作業の低減とモジュールの低価格化が必要であり，設計構造をモジュラーに変化させていく必要がある．あるカテゴリの製品がモジュラー化されると，適正品質の製品を低価格で開発できるようになるものの，モジュラーであるがゆえに製品の価値はモジュールの総体で決定され，統合製品としての差別化や付加価値の付与が難しくなる．このとき企業は競争力の高いモジュールに特化する，アーキテクチャやプラットフォームを押さえる，サービスに特化するなどの回避戦略を採る必要がある．

部品統合 / 部品内	組み合せ	擦り合せ
組み合せ	1. 価格競争領域	3.部品間擦り合せ
擦り合せ	2.部品内擦り合せ	4.高価格ニッチ領域

表-1　組み合せ／擦り合せの組み合せ

しかし，ユーザが要求する統合品質が変化し，より一層の統合品質を必要とする場合には，再度擦り合せが必要になる．モジュールで十分な品質を提供できる場合は，高品質なニッチ市場を狙うことになり，モジュールで十分な品質を提供できない場合は，擦り合せ開発で高い競争力を持つことができる．

このように，組み合せと擦り合せは，製品に対する市場のニーズ，トレンド，成熟度，ユーザが要求する統合品質にあわせて選択されるべきビジネス上の戦略であり，企業が設計現場だけの都合で選べるわけではない．

■組み合せと擦り合せの融合

システム規模が大きくなると，当然，組み合せ戦略が必要になる．部品内と部品統合の2つに分けて整理すると，擦り合せが得意な企業が選択すべき領域は**表-1**中の2と3であることが分かる．

組み合せと擦り合せは依存関係の凝集パターンの違いであるため，組み合せ構造であってもモジュールが高い凝集度を持っていて，擦り合せ開発によって初めて必要とされる品質が維持できるケースがある．これが2の領域である．モジュール化が進んだときに，自社製品の中に価値の高いモジュールを見つけ，高品質なモジュールを複数の製品に提供していくことで競争力を維持する戦略である．この戦略で勝つためには，擦り合せでなければ品質が出しづらいモジュール，モジュール品質の高さが統合システムの価値を引き上げるようなモジュールをいち早く見つけることが条件になる．

これとは反対に，モジュラーな構造ではあっても，L字部分に高い凝集度が要求されるため，擦り合せ開発によって初めて必要とされる統合品質が達成できるケースもある．これが3の領域である．この戦略で勝つためには，組み合せで製品を作ったときに提供できる品質では，標準的なユーザが要求する品質を提供するのが困難であることが条件になる．

■組み合せ力強化

開発規模が小さい場合は，擦り合せ開発が統合品質を出す上で有利であるのは確かであるが，グローバル化に対応

する競争力を持つためには，組み合せ力の向上も欠かせない．

組み合せ力の基本は戦略的な構想力である．日本の組込みソフトウェア業界にはいわゆるアーキテクトと呼ばれる人材が育っていないことが指摘されているが，その一番の理由はアーキテクトという職種がない，つまり，アーキテクチャを構想することが仕事として認められていないことが大きな原因であろう．

擦り合せべったりの企業の場合は，組み合せアーキテクチャを構想するだけでなく，組織文化の修正，プロセスの変更と厳格化，人事制度の修正，組織変更，アウトソースのための設計仕様記述能力，外部組織のプロジェクト管理能力など，組織能力のシフトに包括的に取り組む必要がある．

人材の流動性向上，成果主義，派遣から業務委託への切り替え，海外アウトソーシングの浸透など，組み合せにシフトするための条件は整いつつある．しかし，組み合せへのシフトに取り組んだ企業が必ずしも成功しているとはいえない状況は，企業の包括的な取り組みが不十分であることを表しているようだ．これは，日本企業の擦り合せ文化の根が深いことの示唆でもある．

組み合せへのシフト自体は擦り合せ力を向上させるわけではないため，統合品質を捨ててでもモジュールのオプション価値をベースにしたグローバルな競争力を高めるのが目的である，というトップと現場の意識共有が必須である．

■擦り合せ力強化

日本の企業が総じて擦り合せを得意とするとしても，すべての企業が擦り合せに強いわけではない．TQM活動が浸透していない企業や，十分浸透していてもソフトウェア部門に改善意識が希薄な企業は，擦り合せ力の強化に取り組む必要がある．また，組み合せにシフトするための条件が整いつつあるということは，裏を返すと擦り合せ力が低下する条件が整っているということであり，擦り合せ力を自負する企業であっても油断はできず，より一層の努力が必要である．

組み合せも擦り合せも複雑度に対応する戦略であるが，概して擦り合せ開発では依存度の爆発を放置する傾向がある．モジュラーに移行するのは複雑度を抑えるための手段の1つでしかなく，擦り合せ開発でも依存関係を適切に抑えてコントロールできれば十分効果がある．

現在の擦り合せ型開発の問題点は，後半のシステム統合時の，バグの多発，仕様変更の多発にエンジニアリングで対処できていないことである．実際のところ後半のテストで不具合が続出するというよりも，擦り合せ品質向上活動が本格化する統合作業が後半に集中していると理解したほうが実態を正しく捉えていることも多い．これに対処するためには，シミュレーションやコデザイン手法によるフロントローディングや，ユースケースドリブンの開発手法などを用い，前半からの統合機会を増やし，擦り合せによる品質向上活動を前半へ分散し，後半への集中を防ぐ必要がある．

しかし，統合機会を増やしても勘と経験による旧態依然たる開発では擦り合せ過剰に陥る危険が高い．仕様変更は仮説検証の結果であることを意識して検証項目を客観的な評価基準でコントロールしていくとともに，統合タイミングごとの統合品質目標をコントロールすることで，擦り合せ開発をエンジニアリングにのせていく必要がある．

現在のところ擦り合せ型開発の整理されたソフトウェアエンジニアリング体系は存在しない．今後整備拡充が進むことが望まれるが，多くの工夫は現場での改善活動のベストプラクティスの積み重ねでしかなく，まず現場が勘と経験だけの世界から脱却し，エンジニアリングの意識を持つことがスタート地点である．

戦略的能力構築に向けて

製品の高度化・高機能化によって組込みソフトウェアの開発規模が増大し，品質に重要な影響を与えるようになってきた．組込みソフトウェアの領域でも日本得意の擦り合せ開発が統合品質に貢献していたが，規模増大に伴う複雑さの増大に対して職人的ものづくりでは十分な品質が確保できなくなってきている．現在の日本は擦り合せ過多であり，モジュール化のオプションを持たない企業はグローバルな競争では厳しい戦いを強いられることが予想される．

かといって，苦手な組み合せに完全にシフトした結果，競争力の源であった統合品質と擦り合せ能力を失ってしまっては本末転倒である．いま現在必要なのは，組み合せ・擦り合せの特性を理解した上でのバランスのとれた両面戦略である．高品質な製品を作り出す背景にある競争力を鍛えるには，組み合せ力を強化すると同時に擦り合せ力も強化し，両者をうまく使い分ける戦略的志向能力である．

アーキテクチャはビジネス環境におけるユーザの統合品質に対するニーズと密接に結びついている．現場の設計の論理だけで決めるべきものではない．日本の企業が強固な競争力を持つためには，ビジネス視点と現場視点の両方に基づいたアーキテクチャ戦略と，それを現実の製品品質に結びつける組織能力を鍛える必要がある．組込みソフトウェアにおいても，アーキテクチャ戦略をベースにした競争力強化が重要な時期にきている．

参考文献

1) 藤本隆宏：日本のもの造り哲学，日本経済新聞社（2004）．
2) カーリス・Y・ボールドウィン，キム・B・クラーク：デザイン・ルール―モジュール化パワー，東洋経済新報社（2004）．
3) L. コンスタンチン：ソフトウェア開発のカオス，共立出版（2003）．
4) R. E. ニスベット：木を見る西洋人 森を見る東洋人（2004）．

（平成17年4月13日受付）