RIETI Discussion Paper Series 06-J-040

モジュール製品間の統合は可能か
―パソコン NC の成立過程の分析―

柴田 友厚
香川大学

児玉 文雄
経済産業研究所

RIETI
Research Institute of Economy, Trade & Industry, IAA
独立行政法人経済産業研究所
http://www.rieti.go.jp/jp/

RIETI Discussion Paper Series 06-J-040

モジュール製品間の統合は可能か
―パソコンNCの成立過程の分析―

2006 年 5 月

柴田友厚
香川大学大学院地域マネジメント研究科

児玉文雄
RIETI ファカルティフェロー
東京大学
芝浦工業大学専門職大学院工学マネジメント研究科

要　旨

モジュール統合とは、異なる技術軌跡を辿る２つの製品システムがそれぞれモジュール・アーキテクチャに到達した後に、２つのシステムが統合されて、新たな製品コンセプトが生まれることを意味する。そのような統合はそもそも可能なのか。可能だとすれば、どのようにして実現されるのだろうか。本稿では、パソコンＮＣを事例を分析してモジュール統合の可能性を例証する。この事例研究に基づき、製品アーキテクチャのダイナミクスに関する既存研究の到達点を確認した結果、転用オペレータの作用によって、製品アーキテクチャが再編されることを発見した。この発見を基に、モジュール統合の可能性を理論的に考察する。

１．はじめに

製品アーキテクチャに着目してイノベーションを考えることは、個別技術そのものよりも、製品全体の設計思想に注意を払おうとすることである。製品アーキテクチャというレンズを通して、個別技術の組み合わせ方や関係性がどうなっているのかを見ようとするのである。このような製品アーキテクチャの優劣と個別技術の優劣とは基本的に別物であろう。個別技術が優れていても、必ずしもアーキテクチャが優れているとは限らないし、優れた個別技術を持っていながらアーキテクチャで失敗したために、事業としてはうまくいかないということは大いにありうるのである[1]。

したがって、製品アーキテクチャのダイナミクスに関する理解を深めることは、イノベーションに関する知見を広げることにつながるであろう。既存研究は、製品アーキテクチャとは、インテグラルからモジュールへ次第にシフトすること、また、モジュールからインテグラルへ逆シフトするようなダイナミクスを持つ、ということを明らかにしてきた。本稿では、まず、ＮＣシステムとパソコン・システムの統合を事例にとり、モジュール統合に関する理解を深める。続いて、現時点での既存研究の到達点を確認し、その上で、新たにモジュール統合の可能性を理論的に導出する。このようにして、製品アーキテクチャのダイナミクスに関する知見を更に深めるのが本稿の目的である。ここでモジュール統合とは、異なる技術軌跡を歩んできた２つの製品システムが、それぞれモジュール・アーキテクチャに到達した後に、モジュールレベルでの転用がおこり、製品アーキテクチャが再編されるという現象を意味している。まさに、モジュールの組み換えが産業を超えて起こり、製品アーキテクチャが再編され、組み合わせの妙味によって新たな製品コンセプトが生まれるのである。

---

[1] 例えばアップルコンピュータの例を考えてみる。アップルコンピュータはその優れたＯＳを低品質のハードウエアと一緒に統合して組み込んだ。その結果、ハードウエアが足かせとなり、優れたＯＳを十分に生かすことができなかった。もしＯＳをハードウエアから独立させていれば、ＯＳ単体の優れた特性を十分発揮することができ、事業として成立したかもしれない。これはまさにアーキテクチャの問題であり、個別技術の問題ではない。パソコンという製品システム全体を、どのようなモジュールに分け、その間にどのようなインタフェースを設定するかという問題なのである。ＯＳという個別技術は優れているが、アーキテクチャが原因で事業としてはうまくいかなかった例である。

２．パソコンとＮＣのモジュール統合[2]

現在、欧米のＮＣ工作機械市場では、パソコンの高度な情報処理機能とＮＣの制御機能を統合したいわゆるパソコンＮＣが台頭し、大きなシェアを占めている。たとえば米国ＧＭ向けのＮＣシステムは、現在ではその多くはパソコンＮＣである[3]。本節ではその事例を紹介することで、モジュール統合に関する理解を深める。

パソコンとＮＣ装置は、異なる技術軌道を歩んできた製品であり、産業も異なるり市場も異なる。パソコンは一般ユーザーを対象にした消費財であり、他方ＮＣ装置は、工作機械に付加されて工作機械を自動制御するシステムであり生産財である。市場も顧客も異なるが、製品アーキテクチャには共通点がある。それはパソコンはオープン・モジュールであるのに比べて、ＮＣ装置はクローズド・モジュールという違いはあるが、両者ともモジュール型の製品アーキテクチャを採用しているということであろう[4]。パソコンは、ディスプレイ、マザーボード、キーボードなど複数のモジュールで構成されているが、オープン・モジュールのために、各モジュールは市場から調達することが可能である。他方、ＮＣ装置の場合、表示部、制御部、モータ部などのモジュールで構成されるが、これはクローズド・モジュールであるために、市場からの調達はできず自社内からの調達に限定される。このような環境の中で、パソコンのモジュールをＮＣシステムの表示部に利用することが可能になり、パソコンとＮＣの統合が実現した。

ＮＣのアーキテクチャは、1975 年にＭＰＵ（Micro Processor Unit）を搭載したＮＣ装置が開発されて以降、図表１に示すようにモジュール化が進展してきた。そして 1991 年以降の第３世代ＮＣでは、そのアーキテクチャは、表示部、演算部、駆動部という大きな機能ごとにモジュール化されたが、この分割方法が、パソコンとのモジュール統合を促進した。

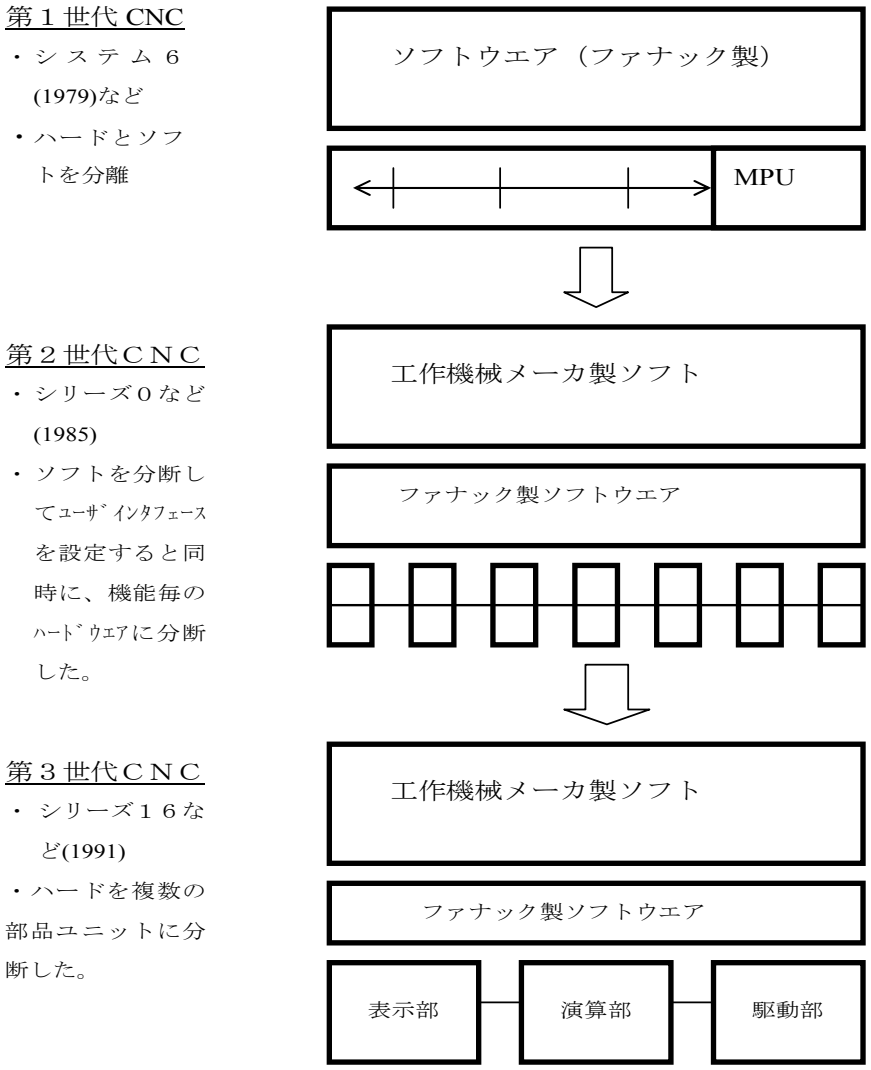
---

[2] 本事例は、ファナック株式会社の製品カタログ、および財団法人製造科学技術センター 岡宗秀一氏へ行なったインタビューをもとにして作成した。

[3] ２００５年９月１５日に財団法人製造科学技術センター 岡宗秀一氏に対して行なったインタビュー

[4] モジュール・アーキテクチャは厳密には、クローズド・モジュールとオープン・モジュールを区別して議論することが望ましい。クローズド・モジュールの場合、モジュール間のインタフェースは企業内部で共有されるにとどまるために、分業の範囲は企業内での分業にとどまる。産業構造にまで大きな影響を与えることはない。他方、オープン・モジュールの場合、モジュール間のインタフェースは社会的に共有されるために、分業の範囲は企業内分業を超えて社会的分業にまで広がる。その結果、産業構造にまで大きな影響を及ぼす可能性がある。典型的な例は、パソコンである。そしてパソコンのオープン・アーキテクチャ化によってパソコン産業全体は活性化したが、しかし、IBM 互換機の台頭により IBM のパソコン事業そのものは次第に利益を上げることが難しくなっていった。

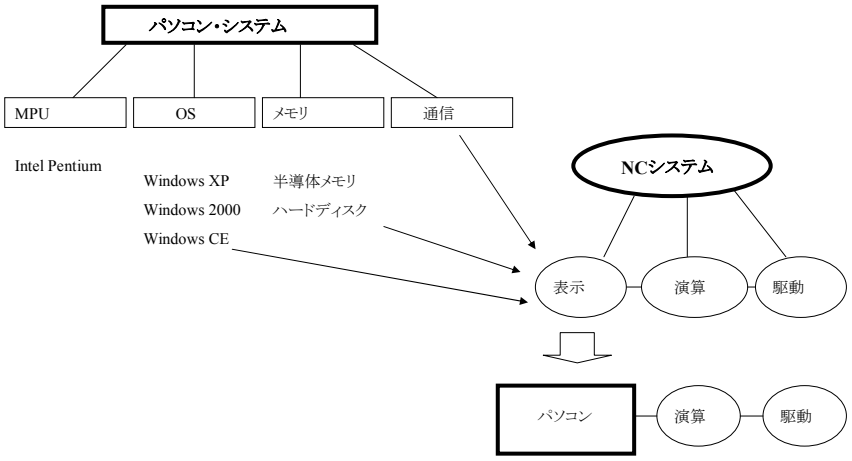
図表１．ＭＰＵ導入後のアーキテクチャ変化の概観図



出典：柴田、玄場、児玉：「製品アーキテクチャの進化論」(白桃書房、2002 年)

表示部、演算部、駆動部というアーキテクチャを最初に実現したＮＣ装置はシリーズ１６である。このアーキテクチャは、プリント板を３次元的に利用するという画期的な３次元実装方式の開発や、電子部品の高密度実装技術などによって可能になった。表示部はまさに文字通り、工具や工作機械の位置表示やＮＣプログラムの表示などを行ない、演算部は、工具の速度や経路を計算して加工に必要なデータを駆動部に転送し、駆動部はそれのデータをもとにしてモータを制御する。そして、それらの３つのユニットの間は、ファナック・シリアルバスという専用インタフェースによって連結されている。このアーキテクチャをＮＣ装置が採用したことによって、表示モジュールへのパソコン機能の利用が低コストで可能になり、図表２に示すようなモジュール統合が実現したのである。

図表2　パソコンとNCの統合



パソコンとＮＣのモジュール統合には、２つの方法が存在する。第１の方法は、市販パソコンをそのまま、表示ユニットに利用する方法である。この場合、ファナックのＮＣはクローズド・モジュールでインタフェースが公開されていないために、市販パソコンとファナックのＮＣでは異なるバスの規格を採用している。そのため、バス規格を相互に変換する必要があり、そのためのインタフェース・ボードを介して市販パソコンとＮＣ装置の演算モジュールを連結する必要がある。演算モジュールや駆動モジュールを修正する必要はなく、インタフェース・ボードの追加のみでパソコンＮＣが可能になるのである。

第２の方法は、インテルＭＰＵ、Windows XP や半導体メモリなどの市販モジュールを市場から調達して、組み合わせてパソコン機能を作り出し、それを表示ユニットに利用する方法である。このようにして作り出したパソコン機能搭載のファナック独自の表示ユニットを PANELｉ と称する。この場合、インタフェース・ボードは必要ない。この方法の場合、パソコンを構成する市販モジュールから、原理的には任意のモジュールを組み合わせて、機械加工現場の厳しい作業条件に適したパソコン機能が搭載可能になる。例えば、高い信頼性を要求する作業環境に備えて、Windows XP の代わりに Windows CE を採用し、ハードディスクの代わりに半導体メモリを採用した PANELi を作ることが可能になる。このようなことは、単に市販パソコンを表示ユニットに利用しただけでは実現不可能であろう。

パソコンの表示モジュールへの利用によって、表示ユニットは、従来の単なるＮＣデータの表示機能だけではなく、ネットワーク機能やデータベース機能などの広汎で柔軟性の高いパソコン機能を、ＮＣオペレータに提供することが可能になった。例えば利用されたパソコン機能の１つであるデータベース機能を利用して、ＮＣオペレータは工具ファイルを管理したり、独自の操作画面を作成してヒューマン・インタフェースを自由に構築することが可能になった。更に、パソコンのネットワーク機能を利用することで、インターネ

ットを介して工場内の離れた場所からＮＣを操作することも可能になった。利用されるパソコン機能によってもたらされるデータベース機能やネットワーク機能は、ＮＣシステムを単に工作機械を制御する装置から、制御データの分析や遠隔操作までが実現できる装置へと変化させた。つまりパソコンの豊富な情報処理機能が制御機能と組み合わさることで、一層高度な価値を提供できるようになったのである。そのような意味において、パソコンの持つ情報処理機能はＮＣの持つ制御機能と相互に価値を高めあう関係になり、情報処理機能との統合によってＮＣシステム全体の価値は更に向上したと考えることができる。

このような統合に要するコストは、システム間の相互依存性の程度と密接な関係があり、相互依存性が高いほど統合に要するコストは高くなる。ＮＣシステムを、表示部、制御部、駆動部という３つのモジュールに分割したことは、相互依存性の問題を生じることなくパソコン機能を表示部に利用することを可能にした。ＮＣシステムにＭＰＵが導入された 1975 年以降、モジュール化は次第に進展してきたが、既に述べたように、最初から表示部、制御部、駆動部という３つのモジュールに分割されていたわけではない（Shibata,Yano and Kodama,2005）。現在の分割方法が実現したのは 1991 年の第３世代ＮＣ以降であり、それ以前はより細分化された機能ごとのハードウエア・モジュールに分割されていた。例えば、通信機能が１つのハードウエア・モジュールであったが、このように細分化されたモジュール化では、一まとまりのパソコン機能をＮＣシステムに容易に利用することは難しい。相互依存性の問題なく、パソコン機能をＮＣシステムに利用するためには、第３世代の分割方法まで待たなければならなかったのである。

３．既存研究

製品アーキテクチャに関する先行研究は多い（Baldwin and Clark,1997；Baldwin and Clark,2000; Langlois and Robertson ,1992 ; Robertson and Ulrich,1998 ; Ulrich ,1995; Sanchez and Mahoney,1996;藤本・武石・青島,2001；Shibata,Yano and Kodama,2005）。しかしその中でも、製品アーキテクチャのダイナミクスに言及した先行研究は限られている。製品アーキテクチャのダイナミクスに関して、これまで何が明らかになってきたのかを整理し、現時点の到達点を確認する。既存研究は異なる３つの視点から、製品アーキテクチャは次第にインテグラルからモジュールへ進化するということを明らかにしている。

第１の理由は、製品に対する顧客の評価基準はそのライフサイクルに応じて変化するが、企業はその変化に対応すべく製品アーキテクチャを適応させようとするからである（Christensen and Raynor,2003）。全ての新製品はその初期段階は性能が低く、顧客の要望を十分に満たす水準に達していない。そのような競争状態の時、企業は製品性能の最適化によって顧客の要望に答えようとするはずであり、したがって、インテグラル・アーキテクチャを選択する。インテグラル・アーキテクチャの方が、モジュール・アーキテクチャよりも製品性能の最適化を実現しやすいからである。しかし、急速な技術進化により、製品性能は顧客の要求水準をオーバーシュートしてしまい、その時点では、顧客の評価基準は単なる製品性能ではなくて、仕様の柔軟性や使い勝手などの他の基準にシフトする。そのような競争環境の時、企業にとって迅速で柔軟な製品開発を可能にする、モジュール・アーキテクチャを選択するほうが合理的である。技術進化の結果、モジュ

ール・アーキテクチャでも、顧客が満足する製品性能を提供することは可能だからである。このよう理由により、企業は製品アーキテクチャを次第にインテグラルからモジュールへとシフトさせる、というのがクリステンセンらの主張である（Christensen and Raynor,2003）。

第２の理由は、サプライヤーとの効果的な関係性を築くには、製品をモジュール化することが有効だからである（Fine、1998）。技術革新の激しい時代においては、全ての製品開発を自社内で行なうことはほとんど不可能であり、したがって効果的なアウトソーシングのマネジメントが１つの鍵である。そのためには、外注しようとする仕事をできるだけ、一まとまりのパッケージにして、外注先と自社との間で、委託業務に関する複雑な相互依存関係をできるだけ排除し、インタフェースをルール化することが効果的であろう。このような努力は、製品のモジュール化を促進する方向に作用する。ＩＢＭが、パソコンのアーキテクチャをモジュール化したのは、まさにこのような理由によっている。効果的なアウトソーシングのためには、サプライヤーとのインタフェースをルール化し標準化したほうが良く、それは製品のモジュール化を促す。

第３の理由は、企業の組織能力という視点である。インテグラル・アーキテクチャを設計するよりもモジュール・アーキテクチャを設計するほうが、より高度な組織能力を必要とするからである（Baldwin and Clark,1997）。なぜならば、モジュール・アーキテクチャは、各モジュール毎に並行して設計した後に、最終的に全てのモジュールを統合して有機的なシステムとして稼動することを検証しなければならない。事前に設定したデザイン・ルールが、うまく稼動するかどうかは、実際に統合してみてはじめてわかる部分が多くある。最後の検証段階ではじめて、モジュール間での予期せぬ相互依存性が明らかになる場合もあり、その場合再設計の必要性に迫られる。それを避けるためには、事前にシステム全体を俯瞰した効果的なデザイン・ルールを設定することが必要だが、それはシステム全体に関する豊富な経験と知識を必要とし、それほど容易なことではない。ＩＢＭが 1964 年にシステム 360 のモジュール化を実現するに際して多くの困難に直面した事実は、そのことを如実に物語っている（Baldwin and Clark,2000）。このように企業の組織能力という視点に立つと、企業はインテグラル・アーキテクチャの設計経験のなかで次第に知識と経験を蓄積した後、はじめてモジュール・アーキテクチャの設計が可能になるはずである。

既存研究はさらに進んで、モジュールへのシフト後、再度インテグラルへ逆シフトする経路の存在を明らかにしている。インテグラルへの逆シフトは要素技術の革新によって引き起こされるが、その革新の契機には、他産業によって外生的にもたらされる場合と、内生的に自産業によってもたらされる場合とがある。前者の事例は、ＮＣ装置にＭＰＵを導入したことによって、クローズド・モジュールからインテグラルへ逆シフトした場合にみることができる。（Shibata,Yano and Kodama,2005）。ＭＰＵは、ＮＣ装置とは製品ヒエラルキーが異なる半導体産業で起こった革新であり、外生的な要素技術の革新と考えることができる。つまり他産業で誕生した革新的な要素技術をＮＣ装置に採用したことによって、製品アーキテクチャがモジュールからインテグラルへ逆シフトしたのである。

後者の事例は、楠木・チェスブローが指摘したＨＤＤ産業におけるヘッドの革新にみることができる（楠木・チェスブロー,2001）。この産業では、フェライトヘッドから薄膜ヘッドへ、薄膜ヘッドからＭＲヘッドへ

というヘッド技術の革新に際して、そのたびに、モジュール型からインテグラル型へと逆シフトしたことが指摘されている。これらの一連のヘッドの革新は、ＨＤＤという同じ製品ヒエラルキーの中で発生した要素技術の革新であり、内生的な技術革新と考えることができる。つまり、ＨＤＤを革新するメカニズムの中に、アーキテクチャをインテグラルへ逆シフトするロジックが内包されているのである。

このように、内生的にせよ外生的にせよ要素技術の革新によって、製品アーキテクチャがモジュールからインテグラルへの逆シフトする場合がある。それは以下の２つの理由による。第１の理由は、革新的要素技術を中心にした新たな技術体系のもとで、当該製品の性能という評価軸が、再度顧客にとって重要な価値に浮上してくるためである。その場合、モジュールよりもインテグラルが全体最適を実現しやすいために、企業は製品アーキテクチャをインテグラルへシフトさせる（Christensen and Raynor,2003）。第２の理由は、モジュール化を実現するために企業が蓄積してきた知識やノウハウが、革新的要素技術の採用によって無効になるからである。革新的要素技術の採用によってサブシステム間インタフェースを新しく仕切りなおす必要がでてくる。このことはモジュール化を実現するために蓄積してきた知識、ノウハウ、経験が一度無効になることを意味している（Shibata,Yano and Kodama,2005）。しかし、新たな技術体系のもとでのモジュール化を実現するためには、企業はまだ十分な知識やノウハウを蓄積しておらず、その結果、企業は再度インテグラル・アーキテクチャを採用せざるを得ない。以上のような２つの理由によって、革新的要素技術が採用された場合、製品アーキテクチャはモジュールからインテグラルへ逆シフトする。

以上が現時点での、製品アーキテクチャのダイナミクスに関する既存研究の到達点である。次節以降では、パソコンとＮＣの統合の事例研究を参照しながら、モジュール統合という新たなダイナミクスの可能性を考察し、製品アーキテクチャに関する理解を更に深める。

４．転用オペレータ

クローズド・モジュールに到達した製品システムは、もちろん全てがインテグラルへ逆シフトするわけではない。一定の条件を満たす場合に、本稿で「モジュール統合」と称する新たな現象がうまれる可能性がある。それは、異なる技術軌道を歩んできた製品間で、産業を越えてモジュールの交換や置換がおこり、製品アーキテクチャが再編され、新しい製品コンセプトが生まれるという現象である。モジュール統合の可能性は、モジュール化オペレータの１つである転用（Porting）オペレータの作用によって、合理的に説明される。

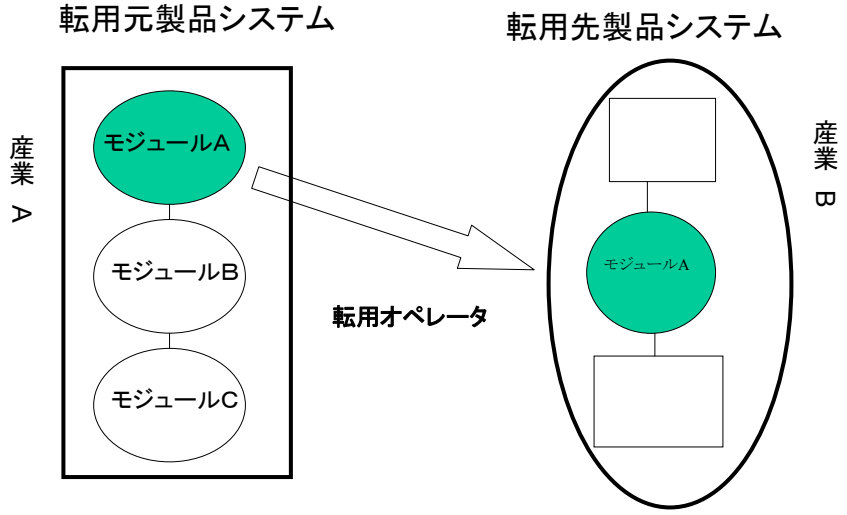
転用オペレータは、モジュール・アーキテクチャに働く６つのモジュール化オペレータのうちの１つである。ボールドウイン＆クラークは、モジュール・アーキテクチャに働く力を、６つのオペレータ、すなわち、分離、交換、追加、削除、抽出、転用という６つのオペレータの作用として概念化し、これらのオペレータが各モジュールに自立分散的に作用することで、モジュール・アーキテクチャの急速な設計進化がもたらされると主張する（Baldwin and Clark,2000）。転用オペレータは、あるモジュールを他のシステムに移植させるように作用する。「システム統合」において転用オペレータに着目するのは、６つオペレータの中では、他のシステムへ作用するものは転用オペレータのみであり、他のオペレータは当該システムの中でのみ作用す

るからである。

転用オペレータが働いた典型的な例としては、モジュール・アーキテクチャだった UNIX オペレーティングシステムが、多数の異なるハードウエア上で移植されて動作するようになった例をあげることができる（Baldwin and Clark,2000）。UNIX は当初、ＤＥＣ社の PDP11 のために設計され、PDP11 の命令セットに依存していた。しかし C 言語という高級言語で書き直されたことと、UNIX の高度なモジュール性によって、PDP11 という特定のハードウエア環境への依存から開放されて、多くのハードウエア上で動作可能になった。ここで今後の議論のために、転用オペレータが特定モジュールに作用して、それが移植された移植先のシステムを「転用先システム」と称し、特定モジュールが元来存在していたシステムを、「転用元システム」と称して両者を区別することにする。つまり、PDP11 を転用元システムと称し、UNIX の移植先を転用先システムと称する。

このような転用オペレータの視点に立脚すれば、モジュール統合とは転用先システムにおいて、転用されたモジュールと他のモジュールがうまく統合されることで、新しい価値が生み出される現象として理解できる（図表３参照）。

図表３　モジュール統合概念図



転用先システムのアーキテクチャが以下に示す３つの条件を満たすときに、モジュール統合が形成されるのである。

第１に、転用先システムは、モジュール・アーキテクチャでなければならない。転用オペレータは、転用元システムがモジュール・アーキテクチャであれば作用する力であり、転用先システムがインテグラルであるかモジュールであるかは問題にならない。だが、転用先システムの特定モジュールを置換する形で、モジュール統合が形成されるために、転用先システムはモジュール・アーキテクチャでなければならない。

第２に、転用されるモジュールと他のモジュールとの間に機能的な補完性がなければモジュール統合のメリットはない。ここで機能的補完性とは、ある機能と他の機能とが相互に価値を強化しあうような関係にあり、それによってシステム全体の価値が高まるような場合をいう[5]。モジュールがシステムに転用されることによって、他のモジュールとの間で相互に機能を補完しあい、それによってシステム全体の価値を一層高めることができる場合に、モジュール統合は成立する。だがこのような機能的補完性がいくら存在しても、統合に要するコストや作業量が膨大であれば、モジュール統合を形成する合理性は失われるであろう。そして統合に要するコストや作業量は、他のモジュールとの相互依存性に大きく依存し、それゆえに次の条件が必要になる。

第３に、転用先システムにおいて、転用されるモジュールと他のモジュールとの間に、相互依存関係が生じないように転用される必要がある。相互依存関係が存在する場合、外部から新たなモジュールを転用することによって、他のモジュールを修正する必要があり、その結果システム全体の信頼性の低下やコストの増加につながる。相互依存性が存在しないために、極めて低コストで新たな製品コンセプトを生み出すことが可能になるのである。相互依存関係が生じないようにするためには、転用先システムが、転用されるモジュールにうまく適合するようなアーキテクチャである必要がある。つまり、転用先システムのモジュールの分割方法は、転用されるモジュールと適合的でなければならない。

これらの３条件がそろう場合、転用オペレータの作用により、従来の産業ピラミッドを越えて安いコストで容易に、新たな価値を持つ製品コンセプトを生み出すことが可能になる。

５．考察

本稿ではまず、パソコンＮＣの事例を紹介し、モジュール統合に関する理解を深めた後、製品アーキテクチャのダイナミクスに関する既存研究の到達点を確認し、モジュール統合の可能性を理論的に考察した。産業を超えたモジュール統合によって、製品システムの信頼性を損なうことなく低コストで高い価値を持った製品コンセプトを生み出すことができる。そのためには、前述した３つの条件、すなわち転用先のシステムがモジュール・アーキテクチャであり、機能的補完性があり、相互依存性がないという条件が必要なのである。アーキテクチャという視点から考えれば、モジュール統合により生まれたパソコンＮＣは、以前と同じモジュール・アーキテクチャとして類型化される。しかし内容的にみれば、情報処理機能と制御機能の「統合」によって以前よりも一段高い価値を持っているという意味において、それ以前のモジュール・アーキテクチャとは内容的には大きく異なったものであると考えることができる。

---

[5] 補完性（Complementarity）とは、一方の存在が他方の存在事由になっているような関係をさす。例えば、コンピュータのハードウエアとソフトウエアは補完的な関係にある。また、青木（1996）によれば、制度や仕組みには制度的補完性(institutional complementarity)が存在するという。そしてその補完性ゆえに、１つの経済システムに存在する多様な制度や仕組みは、その経済システムの強靭さを一層強めていると考えるのである。

このようなモジュール統合は、パソコンとＮＣ以外に他の産業でも起こるのだろうか。モジュール統合の理論的背景にある転用オペレータは、特定の産業や製品にのみ働くわけではない。当該製品システムを「システム依存部」と「モジュール固有部」とに明確に分離することができれば、転用オペレータはモジュール固有部分に作用することができる（Baldwin and Clark,2000）。そして転用オペレータが作用したモジュールは、そのシステム依存部分から切り離されて、つまり当該システムのデザイン・ルールを離れて、別のシステムのデザイン・ルールの中に組み込まれる。もちろんその場合、新たなシステム依存部とモジュール固有部分との間を翻訳する機能が必要になる（Baldwin and Clark,2000）。前述したパソコンＮＣの場合、インタフェース・ボードがその役割を果した。このように転用オペレータそれ自身は、製品や産業の違いを問わず、その製品システムがモジュール・アーキテクチャでありさえすれば作用することができる。

だがモジュール統合とは、２つの異なる技術軌道を辿った製品システム間で起こるために、他の製品システムとの間で機能的補完性と相互依存性の条件を満たすことが必要になる。歴史的にみると、コンピュータとＮＣとの出会いは、1970 年代のミニコンにまでさかのぼることができる。ＮＣの主たる機能の１つは、加工物に関する加工データを読み込み、情報処理をして数値データに変換することであり、その意味において、情報処理に優れたコンピュータ技術を取り込もうとする誘引はＮＣの初期から存在していたということができる。だが、実際にパソコンＮＣが可能になるには、第３世代ＮＣのアーキテクチャが生まれ、パソコンとの間でこれらの条件を満たすまで待たなければならなかったのである。

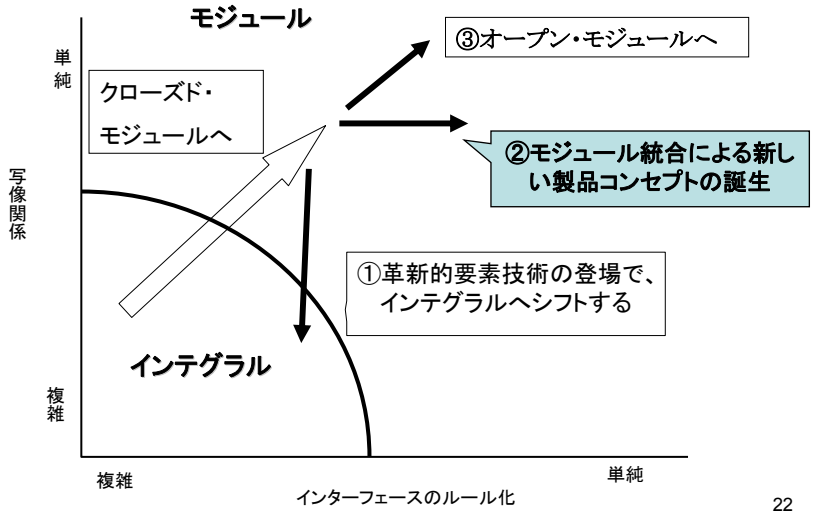
1960 年代後半から 70 年代にかけて、ミニコンピュータいわゆるミニコンが出現した。それは国内外の各社からいくつか発表されたが、たとえば、ＤＥＣ社の PDP-8 や日立の HITAC-10 などがそれである。これらのミニコンの標準的なサイズは、奥行きが数 100mm, 高さ 150 から 100mm の箱のなかに、ＣＰＵやメモリが内蔵されている、というものであった。ファナックは、1972 年にミニコンを内蔵したファナック 200A を世界で始めて開発したが、大変高価なものであり、用途は極めてハイエンドに限定されていた。工作機械に組み込まれるＮＣにとって小型化は非常に重要な性能指標の１つだが、このサイズは、工作機械の制御部分としては何とか使用可能ではあるが、とても魅力的なサイズではなかった。また当時のミニコンはランダム・ロジックで構成されており、メモリも磁気コアと称されるメモリであったために、価格も 1970 年当時で数 100 万円程度にもなった[6]。更にアーキテクチャという視点からみても、当時のミニコンのアーキテクチャは、オープン・モジュールではなかった。当時のコンピュータとＮＣとの間には、モジュール統合が実現する条件はいまだ十分ではなかったのである。パソコンＮＣが実現可能になるには、ＮＣシステムのモジュール化が一層進展した第３世代ＮＣまで待たなければならなかったし、同時に、半導体技術の進歩により安価で占有面積が小さいオープン・モジュール・アーキテクチャのコンピュータの登場が必要であった。このようにＮＣの歴史は、コンピュータの情報処理機能をＮＣシステムの中に取りこもうとする歴史でもあったが、実際にそれが真の意味で実現するためには、パソコンＮＣまで待たなければならなかったのである。

本稿で展開したモジュール統合の議論は、製品アーキテクチャのダイナミクスに関する理解を広げてくれ

6　「NC システム事典」（1983）

る。既存研究の知見にモジュール統合の議論を組み込むと、製品アーキテクチャのダイナミクスに関する理解を、図表４に示すように拡張することができるだろう。

図表4:製品アーキテクチャのダイナミズム



まず第１に、製品アーキテクチャは、次第にインテグラルからクローズド・モジュールの方向へシフトする。その後は、２つの可能性が存在する。第１の可能性は、画期的な要素技術の台頭により、それを中心にした新たな技術体系で再度インテグラルへ逆シフトする経路である。そして本稿で議論した第２の可能性は、モジュール化した製品システムがある一定の条件を満たす時に、他の製品システムとの間でモジュール統合を形成し、それによって製品アーキテクチャが再編されるという経路である。再編された製品システムはモジュール・アーキテクチャという意味では同じだが、しかし内容的には従来のシステムより一段価値の高いシステムに到達しているのである。

そしてさらに、クローズド・モジュールに到達後、製品アーキテクチャは実はもう１つの新しい経路を辿る可能性がある。それはクローズド・モジュールからオープン・モジュールへとシフトする経路の可能性である。クローズド・モジュールのままであれば、モジュール間のインタフェースは企業内部で共有されるにとどまるために、分業の範囲は企業内での分業にとどまる。産業構造にまで大きな影響を与えることはない。他方、クローズド・モジュールからオープン・モジュールにまでシフトすると、モジュール間のインタフェースは社会的に共有されるために、分業の範囲は企業内分業を超えて社会的分業にまで広がる。その結果、産業構造にまで大きな影響を及ぼす。また、オープン・モジュールにまで到達した段階では、製品アーキテクチャを所与として、各モジュール毎に競争が展開されるために、この段階で製品アーキテクチャを変更することは非常に大きなスイッチング・コストを要する。その結果、この段階ではアーキテクチャを変革する

ような大きな革新は起こりにくく、現在のアーキテクチャを維持しようとする力学が働く。その意味において、オープン・モジュールに到達した段階とは社会的に安定的な状態だということができるだろう。

それではどのような条件の時に、クローズド・モジュールからオープン・モジュールへの移行が実現するのだろうか。製品アーキテクチャがクローズド・モジュールからオープン・モジュールへまで移行するには、２つの可能性が存在する。第１は、ＩＳＯなどの国際標準化活動によりいわゆるデジュリ・スタンダードが形成されることでオープン・モジュールへ移行するという経路である。第２は、市場競争によりデファクト・スタンダードが形成されることで、オープン・モジュールへ移行する経路である。いずれの場合しても、市場を支配しているキープレーヤーの行動に大きく依存する。パソコンのオープン・アーキテクチャ化は、コンピュータ産業を支配していたＩＢＭが、パソコンのインタフェースを全て公開することによって、それがデファクト・スタンダードになったことにもたらされた。ＩＢＭというコンピュータの巨人の規格だったからこそ、それが公開されるとそれが一気にデファクトになったのである。そしてパソコンのオープン・アーキテクチャ化によって、パソコン産業全体は活性化したが、しかし、IBM 互換機の台頭によりＩＢＭのパソコン事業そのものは次第に利益を上げることが難しくなっていった。このような歴史的事実を考えると、個別企業にとっては、クローズド・モジュールの製品アーキテクチャをあえてオープンにすることにメリットを見出すことは難しいと言えるだろう。

以上のような議論をベースにすると、製品アーキテクチャのダイナミクスの問題は、図表４に示した経路のうちどの経路を辿るのか、それがどのような要因によって規定されるのかということが、明らかではないということである。つまり、製品アーキテクチャの経路に影響を与える規定要因にはどのようなものが存在するのかがまだ十分明らかではなく、これらに関しては更なる研究の深化が必要である。

６．結語

本稿はパソコンＮＣの事例を取り上げたが、本稿で提示した製品アーキテクチャのダイナミクスは、理論的には特定の産業や製品に限定されるわけではない。システムでありさえすれば適用可能であり、一定の汎用性が存在する議論のはずである。更なる理論的発展とその実証は今後の課題としたい。

参考文献

青木昌彦・奥野正寛編著(1996)「経済システムの比較制度分析」、東京大学出版会

Baldwin , CarlisssY, and Kim B. Clark (1997), "Managing in the Age of Modularity", Harvard Business Review, Vol.75 , No.5.

Baldwin , CarlisssY, and Kim B. Clark (2000), Design Rules: The Power of Modularity, Vol.1, Cambridge, MA: MIT Press.

Christensen, Clayton (1998), The Innovator's Dilemma, Harvard Business School Press, Boston, Massachusetts.

Christensen, Clayton and Raynor(2003), The Innovator's Solution, Harvard Business School Press, Boston, Massachusetts.

土井康弘、本多庸悟、井上久仁子編集（1983）『NC システム事典』 朝倉書店.

Fine, Charles, 1998, ClockSpeed: Winning Industry Control in the Age of Temporary Advantage, Reading, Massachusetts: Perseus Books.

藤本隆宏・武石彰・青島矢一編（2001）『ビジネス・アーキテクチャ』有斐閣.

Franke, Nilolause and Von Hippel, E(2003),"Satisfying heterogeneous user needs via innovation toolkits:the case of Apache security software", *Research Policy* 32, pp.1195-1215.

Henderson, R and K. Clark (1990), "Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms". *Administrative Science Quarterly, March,* pp9-30.

Iansiti, Marco and Jonathan West (1997), " Technology Integration : Turning Great Research into Great Products", *Harvard Business Review,* May-June.

楠木・チェスブロー（2001）「製品アーキテクチャーのダイナミック・シフト」 藤本・武石・青島編『ビジネス・アーキテクチャー』有斐閣.

Raghu Garud and Arun Kumaraswamy (1993), " Changing Competitive Dynamics in Network Industries：An Exploration of Sun Microsystems's Open Systems Strategy", *Strategic Management Journal,* July.

Richard N.Langlois and Paul L.Robertson (1992), "Networks and Innovation in a modular system: Lessons from the microcomputer and stereo component industory", *Research Policy*, 21, pp297-313

Robertson, David and Ulrich, Karl (1998), " Planning for Product Platform", *Sloan Management Review*, Summer.

Sanchez, R. and J. Mahoney (1996) " Modularity, flexibility, and knowledge management in product and organization design", *Strategic management journal*, Vol. 17 (Winter special Issue), pp. 63-76.

柴田友厚・玄場公規・児玉文雄(2002)『製品アーキテクチャの進化論：システム複雑性と分断による学習』白桃書房

Shibata,Tomoatsu,Yano Masaharu,Kodama Fumio “Empirical analysis of evolution of Product Architecture”, *Research Policy*, Vol.34(2005),pp.13-31.

Simon, HA (1981), *The Science of the Artificial(2nd Edition)*, Cambridge, Mass; MIT Press(稲葉元吉、吉原英樹訳　『新版　システムの科学』、パーソナルメデア社、1987).

Ulrich, Karl (1995), "The Role of Product Architecture in the Manufacturing Firm", *Research Policy*, 24, pp419-440.