MMRC
DISCUSSION PAPER SERIES

No. 332

設計進化のダイナミクス:
複雑システムのアーキテクチャ研究の流れ

兵庫県立大学
立本　博文

2010 年 12 月

東京大学ものづくり経営研究センター
Manufacturing Management Research Center (MMRC)

http://merc.e.u-tokyo.ac.jp/mmrc/dp/index.html

Revealing dynamics of design evolution:

Literature survey on architectures of complex systems

Hirofumi Tatsumoto, Associate Professor

School of Business Administration, University of Hyogo

Dec. 2010

ABSTRACT

This literature survey on the architectural studies explores the relationship between architectures of complex systems and forms of division of labor. The architectural studies started in the 1960s, focusing on the dependencies among design elements to understand the features of complex systems.

With increasingly complex systems, studies in management of technology introduced the architecture concept to reveal the innovation process of complex products. Complex products have a hierarchical structure that is consisted from upper to lower levels. Innovation in the upper level is defined as architectural innovation, and that in the lower level is defined as component innovation.

Studies in the early 1990s demonstrates that architectural innovation strongly affects a firm's competitiveness while component innovation does not, and that architectural innovation often causes established firms to be overwhelmed by new ones. The major explanation for the failure of existing firms is the lack of organizational recognition. Existing firms tend to ignore architectural changes because their organizational structures are adapted to handle complex system effectively, but lack flexibility.

The following studies in the mid 1990s found noticeable difference between two types of architectures, namely integral and modular, among complex systems. Integral architecture has complex independencies among modules, and modular architecture has simple ones. The comparative studies examined that the two architectures differently affect innovation process, emphasizing the different keys to success: Organizational integration for integral architecture and industrial standardization for modular architecture.

Since the two architectures have opposing characteristics, architectural change from integral to modular makes a severe impact on industrial structure and competitiveness. Many researchers focus on this architectural dynamics, investigating how the two main dynamic processes, design evolution and standardization, lead to changes in architecture. Further researches expect to determine effective strategies in the dynamic process of architectural changes.

Keywords：complex system, architectural innovation, hierarchy, integral and modular, design evolution

設計進化のダイナミクス:
複雑システムのアーキテクチャ研究の流れ

兵庫県立大学 立本博文

2010 年 12 月

要約： 「人工物の構造」と「分業構造」の関係を研究するアーキテクチャ研究について文献サーベイを行った。アーキテクチャ研究では、設計要素間の結合関係を人工物の特徴として取りあげている。この考え方の源流は 1960 年代にまで遡る。

1990 年代初めに複雑な人工物を扱う産業のイノベーションパターンに対してアーキテクチャ概念を適用しようという研究が始まった。ここで「下位階層のイノベーションは競争構造に影響を与えない。一方、上位階層でのイノベーションは既存企業が新規企業に打ち負かされるような大きな影響を競争構造に与える」事が分かった。また既存組織はアーキテクチャレベルのイノベーション（上位階層のイノベーション）に対処する事が難しい事も分かった。

1990 年代半ばには、アーキテクチャのタイプによって競争構造に影響を与える経路が異なる事が判明した。モジュール間の結合関係が複雑なインテグラルアーキテクチャでは組織間統合が成功要因である。一方、モジュール間の結合関係が簡明なモジュラーアーキテクチャでは、組織間統合は重要ではない。かわりに標準化を主導しながらネットワーク外部性を利用して競争優位を構築したり、自らに有利な産業構造を創り出したりすることが成功要因であることが指摘された。

2000 年代にはいるとアーキテクチャ・タイプの比較という静的研究から、動的プロセスに研究の焦点が移った。インテグラルアーキテクチャからモジュラーアーキテクチャへの変化は大きな産業変化を引き起こすため、この動的プロセスに対して研究者の関心が集まっている。このアーキテクチャ変化では、設計進化プロセスと標準化プロセスという 2 つの動的プロセスがお互いに影響を与えながらアーキテクチャの進化経路を決定づける。モジュラーアーキテクチャ下ではこの 2 つのプロセスを活用する事が重要な競争戦略となるため、更なる研究が期待されている。

キーワード：複雑なシステム、アーキテクチャルイノベーション、階層性、
インテグラルとモジュラー、設計進化

目次

1 節　なぜアーキテクチャ研究なのか？

アーキテクチャ研究とは

「人工物の構造」と「分業構造」の関係を明らかにする研究をアーキテクチャ研究と呼ぶ。アーキテクチャ研究は技術経営分野では中心的な研究テーマの一つであり、人工物を扱う多くの領域（例えば製品開発論、生産管理論、組織間関係論、産業組織論、企業戦略論）に影響を与えている。現在ではアーキテクチャ概念に対する考察が深まり研究が精緻化し、複雑な議論も増えてきている。しかしもともとアーキテクチャ研究は、非常にシンプルな発想と驚くべき考察に端を発している。

「アーキテクチャ」と言う言葉を人工物設計の視点から初めて用いたのは Simon(1962)である。アーキテクチャとは、簡単に言えば、「設計要素間の結合状態」のことである。つまり、アーキテクチャ研究とは「人工物の特性として、要素間の結合状態を取りあげよう」という非常にシンプルな発想から始まっている。

要素間の結合はどんな人工物も持っている特徴である。だから、どんなものにも「アーキテクチャ」は存在する。実際、Simon(1962)は、アーキテクチャを持っている人工物の例示として、精密機械（時計など）、企業組織（組織構造）、人体などの生体システム（脳や器官など）、記号システム（音符や数式）を挙げている。経済システムやデジタル制御システムもアーキテクチャを持っている。アーキテクチャ研究の古典である Alexander(1964)は、都市設計や文化にもアーキテクチャが存在するとしている。われわれはさまざまなアーキテクチャに囲まれて生活しているわけである。だからアーキテクチャ研究は非常に広い適用可能性をもっており、研究する価値が高い。

ところでアーキテクチャ研究が重要であるとしても、なぜこれほど研究者の興味をそそるのであろうか。それは、アーキテクチャ研究が「複雑なシステムは共通したアーキテクチャをもっている」という仮説を持っているからである。われわれのまわりには多くのアーキテクチャがあるにもかかわらず、複雑なシステムはアーキテクチャ的に共通した特徴があるというのである。

Simon(1962)によれば、精密機械や企業組織、さらには生体システムに至るまで「複雑なシステム」であれば、共通のアーキテクチャ、すなわち共通の設計要素結合パターンをもっている。この驚くべき考察が、アーキテクチャ研究の大きな動機になっている。同様の考え方は、一般システム論研究やサイバネティクス研究(Wiener, 1948)にも見られる。たとえばサイバネティクス研究では、フィードバックやホメオスタティス（恒常性）と言った特徴が持続的な複雑システムに共通していると主張している。（それぞれの研究は、人工物の結合構造に主眼を置くか、そこを流れる情報のフロー・

パターンに主眼を置くかで若干の違いがある。）藤本(2009)は、この考え方の延長にあるものであり、経済システム、精密機械の制御、公理的な設計方法が、複雑な人工物に共通した特性を前提にしていることを紹介している。複雑なシステムに共通する特性を探究することは、長い間、様々な研究者の関心を捉え続けてきたのである。

本稿では、第２節でアーキテクチャ研究の流れを概観する。そこでは初期(1960 年代)のアーキテクチャ研究の理論的なコンセプト、1990 年代前半の階層構造を考慮した研究、さらに 1990 年代半ば以降のモジュール結合構造（インテグラルアーキテクチャとモジュラーアーキテクチャ）の違いによる影響の研究を説明する。そして、2000 年以降、動的プロセス研究へと焦点が移ってきていることを紹介する。

第 3 節では、動的プロセス研究に必至なキー概念と、研究のフレームワークを説明する。そして、動的プロセス研究として設計進化プロセスと標準化プロセスを明らかにする研究に関心が集まっている点を紹介する。

2 節　複雑な人工物のアーキテクチャ

前節では「複雑な人工物が共通のアーキテクチャをもっているのではないか」という考え方がアーキテクチャ研究の動機であることを紹介した。それでは、①なぜ複雑な人工物は共通したアーキテクチャを持っているのであろうか。さらに、②複雑な人工物はどのような共通したアーキテクチャを持っているのであろうか。まず初期のアーキテクチャ研究(Simon, 1962)の考え方をもとに①②について簡単に説明しよう。

2.1　初期のアーキテクチャ研究

複雑な人工物が共通のアーキテクチャを持つ理由

複雑なシステムが共通したアーキテクチャをもつ理由は 2 つある。 1 つめの理由は、人間の認識力の限界である。2 つめの理由は、複雑なシステムが生成されるプロセスにある[1]。

まず認識力の限界が共通したアーキテクチャを生むことを説明する。奥野・瀧澤・渡邊(2006)に見

[1] Simon(1962)は「設計進化プロセス」について詳しく説明しているが、「人間の認識力の限界」についてはあまり説明していない。

るように、人間は限られた認識力しか持たない。そのため、いくつかの関連のある要素を「かたまり」として捉え、複雑なシステムをいくつかの「かたまり」の集まりであると考えている。

たとえば人体を把握するときに、「頭」「体」「手足」というように、ある程度の「かたまり」を認識した後に、「目、耳、口、脳、頭蓋骨で構成されるものが頭だ」というように我々は認識しているはずである（奥野・瀧澤・渡邊(2006)では、複雑システムをいくつかの関連のある設計要素のかたまり毎に分けて考えることを「コーディネーションシステム」と呼んでいる)。たとえば「人体は頭、体、手足で構成される」という風に認識すること無しに、いきなり「人体は神経細胞、筋細胞、上皮細胞で構成される」とは考えないのである。

人間の認識力には限界がある。だから、人間の体は実際には多くの細胞によって構成されているにもかかわらず、いきなり「人体は細胞で構成されている」とは考えない。もしも神経細胞について認識しようとすれば、まずは「手足」つぎに「親指」そして最後に「神経細胞」というように、階層的に認識する。そして各階層は、「頭（目、耳、口等)」「体（胸、腹等)」「手足（親指、腕等)」のように、設計要素をまとめた「かたまり」で構成されている。

つまり、複雑システムは、設計要素の集合である「かたまり」と、「かたまり」から構成される「階層」をもつものとして認識される。これが複雑システムに共通のアーキテクチャである（より正確に言えば、これが複雑システムのアーキテクチャに対する人間の「認識方法」である。)。「かたまり」は現在の用語ではモジュールとよばれている。人間の認識力に限界があるゆえに、複雑システムのアーキテクチャは「モジュール」と「階層構造」として認識されるのである。

ここまで人間の認識力の限界が共通のアーキテクチャを生むという説明をした。つぎに、これとは別の視角、すなわち複雑なシステムが生成されるプロセスから、共通のアーキテクチャが生まれることを説明する。

複雑なシステムは、多くの設計要素をもつ。だから、１つ１つの設計要素から最終的に「複雑なシステム」が完成するまでには多くの時間がかかる。この過程は、複雑システムが完成することを阻害するような多くの邪魔（ノイズ）に満ちあふれている。もし複雑システムが完成したとすると、その過程上には、いくつかの中間的な状態が存在するはずである。なぜなら複雑システムが完成する途中で、邪魔が入ったとしても、もし中間的な状態があれば、その状態から複雑システムを組み上げる事を再開する事が出来るからである。そして中間的な状態から再開する方法の方が、一から設計をやり直すよりも、高い確率で複雑システムを完成させる事が出来るはずである。

この様子を Simon(1962)は時計製造のプロセスで説明している。時計は様々な部品から成り立っている精密機械である。時計組立のプロセスで、時計職人は、時計針、ゼンマイ、歯車などを組み合わせたサブアッセンブリーという「中間状態」をいくつも作る。

もしもサブアッセンブリーなしに、時計を組み立てようとしたら、どうなるだろうか。時計組立中になんらかの邪魔が入ったら、時計職人はもう一度はじめから時計組立をやり直さなくてはいけないかもしれない。それよりも、まず、サブアッセンブリーをいくつか作り、そのサブアッセンブリーをいくつか組み合わせて時計を完成させた方が、はるかに確実に時計組立を行う事が出来るだろう。

複雑システムが完成するためには「中間状態」が有用である。中間状態では、いくつかの関連ある設計要素を「モジュール」として作成するはずである。時計組立の例ではサブアッセンブリーというモジュールを作った。つまり、複雑システムが生成されるプロセス中に「中間状態」を設けると、人工物のアーキテクチャは「複数の設計要素をあつめてモジュールにする」段階と、「モジュールを組み合わせて複雑システムを構成する」段階に、分割される。つまり、複雑システムのアーキテクチャは「モジュール」と「階層構造」という特性を持つのである。

結局、「複雑システムに対する人間の認識力限界」もしくは「複雑システムの生成プロセス」のいずれの説明を採用するにせよ、複雑システムのアーキテクチャは「モジュール」と「階層構造」という共通特性を持つと考えられるのである。

準分解システム

以上のような考察から複雑な人工物（システム）のアーキテクチャが共通してもつ特性を整理すると、①－③の特徴を持つ。これらの特徴をもつ複雑システムを準分解システムとよんでいる（Simon, 1962）。

① モジュール内の設計要素間の依存関係は強く、モジュール間の設計要素間の依存関係は「非常に弱い」もしくは「存在しない」
② 各モジュールの動きは短期的（近似的）には、他のモジュールの動きとは独立している
③ 各モジュールの動きは長期的（集合的）には、他のモジュールに影響する

①は、複雑システムが階層構造を持つことの言い換えである。複雑システムが複雑だとは言っても、全ての設計要素同士が密接に絡み合っているわけではない。設計要素間にほとんど結合がない部分がある一方で、非常に強い結合がみられる部分も存在する。「設計要素間に結合がない領域がある」と言うことは、システムを階層的にとらえることができるということである。

②は、複雑システムの動きが、短期的には各モジュールの動きをすべて集めれば近似できることを意味している。各設計要素をモジュールとして扱うことで、複雑システムは、短期的には、「各モジュールの集合として扱ってよい」と考えられる。

しかし、③の条件は、複雑システムが、各モジュールの合算で類推されるような単純な挙動をしないことを示している。すなわち、長期的には「各モジュールを集合させると、各モジュール間の相互作用が予想外に大きく影響し、システム全体に影響する」ことを示している。

2.2 複雑な人工物の産業に対するアーキテクチャ概念の適用（1990 年初頭の研究）

アーキテクチャの階層性

1990 年代初頭のアーキテクチャ研究では、特に準分解システムの①の特徴が取りあげられた（Clark, 1985; Henderson and Clark,1990; Christensen, 1992a;1992b; Henderson and Cockburn, 1994）。これらの研究では複雑なシステムは、完成品という「上位階層」と部品という「下位階層」に分別された。上位階層は、全体知識（モジュール間の結合に関する知識）が必要なアーキテクチャレベルである。下位階層はコア知識（モジュール内の要素間結合に関する知識）のみが必要なコンポーネントレベルである。

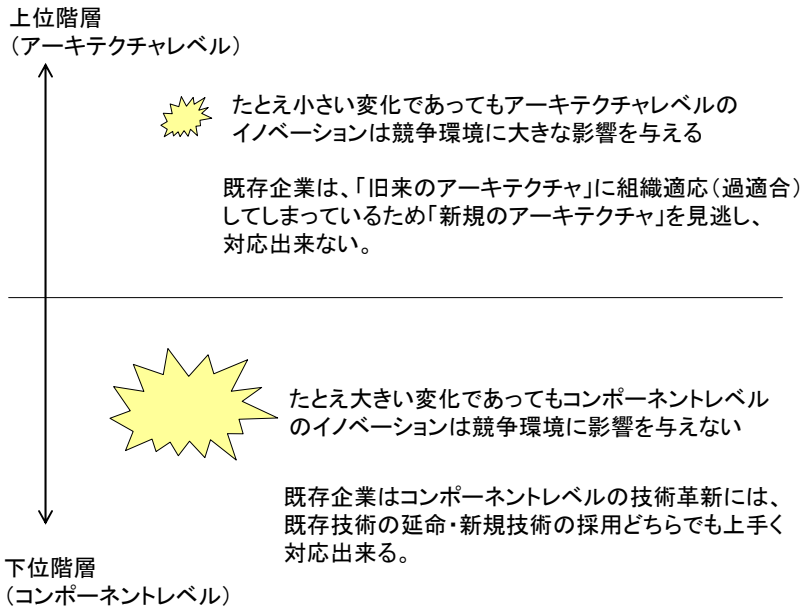

図 1 階層毎のイノベーションの性質

アーキテクチャレベルのイノベーションとコンポーネントレベルのイノベーションが異なる性質を持つことをはじめて指摘したのは、Henderson and Clark(1990)である。彼らは半導体製造装置産業のケース研究を行い、「既存企業はアーキテクチャレベルのイノベーションが発生すると、たとえそれが小さなものであっても、上手く対応する事が出来ず新規企業に打ち破られる」という事実を発見した（図 1)。彼らは、その理由を組織構造が既存のアーキテクチャに適した情報処理構造を持ってしまうため、新規のアーキテクチャの変化を上手く捉えることができない（無視しやすい）ためであると説明した。すなわち組織的な認識力の限界であるとしたのである。

この考えを裏付けるために、Henderson and Cockburn(1994)では製薬産業の研究開発生産性データをつかって実証分析をおこなった。それによれば、アーキテクチャレベルのイノベーションに対して、意識的に組織的対応を行っている企業は研究開発の生産性が高いことが実証された。

この研究結果は 1970-1990 年代のハードディスク産業を調査した Christensen(1992a, 1992b)とも一致した。この研究によれば、コンポーネントレベルで新技術が導入されたとしても、既存の競争構造に対する影響は少ない。既存企業は既存の要素技術の延命でも、新技術導入でもどちらの選択肢

でもこの変化に上手く対応する事が出来ることが観察された。つまり既存企業にとってコンポーネントレベルの技術革新はそれほど脅威ではなかったのである。

しかしアーキテクチャレベルのイノベーションとなると事態は一変する。Christensen は 1970-1990 年代のハードディスク産業では 5 度のアーキテクチャレベルのイノベーションがあったが、そのたびに既存企業は新規企業に打ち破られていったと報告した。この失敗の理由を Christensen(1997)は組織的認識の問題だとしている。

アーキテクチャレベルのイノベーションは新しい製品セグメント（たとえば同じハードディスクでもメインフレームとデスクトップパソコンはセグメントが異なる）の創出を意味しており、そこには異なる顧客選好が存在する。そして往々にして、アーキテクチャレベルのイノベーションは「低価格」「低性能」であり、既存顧客ではなく新規顧客に好まれる。既存顧客の声に誠実な既存企業は、新規顧客の声を無視しやすい。またアーキテクチャレベルのイノベーションがもたらす製品は、既存企業にとって既存製品よりも「低利益」「低信頼」である事が多い。このため既存企業は、アーキテクチャレベルのイノベーションが起こったとしても「合理的に無視してしまう（組織的な対応がとれない）」ために、新規企業に打ち負かされてしまうのだと結論づけ、これをイノベーションのジレンマと呼んだ。この問題は、Henderson and Clark(1990)が指摘した「組織的な認識力の限界」と広い意味で同じ現象である。

アーキテクチャルイノベーションと組織能力との関係

複雑なシステムではアーキテクチャレベルのイノベーション（アーキテクチャルイノベーション）が重要な意味をもっており、これに組織的に対応する事がパフォーマンスに結びつくという考え方は、1990 年代前半の製品開発研究に共通して見られる。Clark and Fujimoto(1991)は日米欧の自動車企業の開発プロジェクト（約 30 プロジェクト）を比較研究し、製品開発生産性が高いプロジェクトではアーキテクチャレベルのイノベーションに対して組織的な対応（組織能力）を行っている点を明らかにした。ここでいう組織能力とは、具体的には重量級マネージャーによる知識統合や、工程間のオーバーラップによって前後工程の知識交換を促すという対応である。

さらに同一企業内だけでなく、企業間（組織間）関係においても、同様のメカニズムが存在する点が報告された。Clark(1989)は、自動車企業と自動車部品企業との取引パターンにも注目し、各国毎に取引パターンが異なる点、さらに取引パターンの差が自動車の開発生産性に影響している点も指摘した。組織間関係で密接なコミュニケーションを行う傾向は特に日本の自動車産業に見ること

が出来た。この能力を関係特殊的資産(Williamson, 1979)や関係特殊的能力(Asanuma, 1989)とよび、アーキテクチャレベルの問題解決に有効である点が強調された。その後の研究によって、自動車産業では組織間関係に関係特殊的能力が存在し、それがプロダクションネットワーク全体の開発効率に大きく影響すること（Dyer and Nobeoka, 2000）、さらにその関係特殊的能力は個々の組織がもつ統合能力と正に相関していること(Takeishi, 2002)が実証された。

これらの一連の製品開発研究は、「複雑なシステムではアーキテクチャレベルのイノベーションが重要であり、これに組織的に対応する事がパフォーマンスに結びつく」という考え方を強くサポートしている。

まとめると、1990 年前半の研究では i)コンポーネントレベルよりもアーキテクチャレベルのイノベーションが産業構造に大きな影響を与える点、ii)アーキテクチャレベルの問題解決には工程間オーバーラップや横断組織、重量級リーダー（中央集権的リーダー）、組織間の関係特殊的能力などの広範な知識を統合する組織能力が重要である点が強調されたのである。

ただし、これらの研究は Simon が主張した準分解システムが持つ特徴①－③のうち、①の点にのみ焦点を当てており、②③の点については焦点を当てていないことに注意が必要である。言い換えれば、「モジュール間の結合パターンによっては組織間統合よりも重要な成功要因があるのではないか」また「モジュールレベルでの技術革新が、どのように集合的にアーキテクチャレベルで影響するのか」といった点は、考察の対象外となったのである。前者については 1990 年代半ば以降のアーキテクチャ・タイプの違いを扱った研究、後者については 2000 年以降のアーキテクチャの動的プロセス研究へと発展していった。

2.3　アーキテクチャ・タイプによる組織間関係の違い（1990 年代半ばの研究）

インテグラルアーキテクチャとモジュラーアーキテクチャ

1990 年代半ばになると、準分解システムの②③の特徴に焦点を当てた研究が増加した。モジュール同士の結合の様子によって、組織に必要とされる能力が異なるのではないかと考えられたのである。

この 1990 年代半ばの研究上の変化は、アメリカ産業のイノベーションパターンが変化した事を反映していると思われる。1990 年代のアメリカ産業の特徴とは、①特定の産業（たとえば IT 産業）、とくにモジュラーアーキテクチャ型の産業にイノベーションが集中している点、②既存企業ではなく新規企業によってイノベーションが主導される点③企業内の資源の活用ではなく、企業外部の資

源活用（産業標準、産学官連携、アウトソーシング等）によってイノベーションが促進されている点が挙げられる（宮田, 2001）。これらの特徴は、アメリカ産業に伝統に見られた垂直統合型企業や企業中央研究所によるイノベーション促進とは大きく異なる。1990 年代に起こったリニアイノベーションからオープンイノベーションへの転換が、一連の研究に大きな影響を与えていると思われる。

アーキテクチャ研究の分野から、この点について初めて言及したのは Ulrich(1995)である。彼はモジュール間の結合状態が簡明なアーキテクチャをモジュラー、複雑なものをインテグラルと区分した。アーキテクチャがモジュラーな状態な場合、複数のモジュールを組み合わせたとしても、集合的な影響は大きくない。だから組織統合も重要ではない。一方、アーキテクチャがインテグラルな場合、集合的な力は予想外に大きくなる。この場合、組織横断的な横串チームや全体統括する重量級マネージャーが必要だと彼は主張した。つまり、アーキテクチャが異なる場合、異なる組織構造が必要ではないか、と問題提起したのである。

同様の考察は組織間関係（企業間関係）の研究分野でも行われた（Langlois and Robertson, 1992; Robertson and Langlois, 1995）。Langlois and Robertson(1992)はステレオ産業やマイクロコンピュータ産業（パソコン産業）を題材に、これらの産業でどのように組織間で調整が行われているのかを検討した。これらの産業では標準規格（互換標準: compatible standards）が存在するため、組織間の調整が極めて簡単に済む。だから濃密なコミュニケーションや特殊な取引パターンといった「関係特殊的能力」なしに企業間の調整を行う事が出来る。

たとえばステレオを考えたとき、互換標準が広く産業に浸透しているため、ユーザーは「スピーカー」「アンプ」「プレイヤー」をさまざまな企業が自由に購入し、組み合わせて 1 つのステレオを組み上げる事が出来る。つまり中央集権的なサプライチェーンではなく、より自律的なサプライチェーンで製品を完成させることが出来る。そして、今後重要となるのはこの自律的なサプライチェーンの下で行われるイノベーション（autonomous innovation）であると Langlois と Robertson は主張した。

Baldwin and Clark(2000)は、モジュラーアーキテクチャがデジタル産業で主流になっている現状をとらえ、モジュラーアーキテクチャ下における製品開発のパターンが、新しいタイプの産業レベルのイノベーション（modular cluster innovation）を引き起こしていると主張した。この代表例がシリコンバレーのコンピュータ産業である。

モジュラーアーキテクチャ下では、各企業は「見える化された設計ルール（visible design rules）」を共有することによって、組織間調整をほとんど行わなくても、様々な部品を組み合わせること(mix

and match)が出来る(Baldwin and Clark, 2000)。「見える化された設計ルール」とは産業標準規格といっても良いだろう。組合せ可能性が広がることによって、製品の潜在的付加価値は爆発的に拡大する。さらに産業全体で並行的な開発（部品専門企業は得意な部品を他の組織と調整無しに開発できる）が実現するため、圧倒的な速さでイノベーションが行われる事を強調した。

「複雑システムを扱う産業にとって、どのような組織間調整（組織統合と分業）が望ましいのか」という問いは、取引コスト理論をベースとした研究からも行われた。そして、この分野でも、1990年代半ばにアーキテクチャのタイプが異なると企業間関係のパターンが異なるのではないか、という見解が提出された(Chesbrough and Teece, 1996)。

もともと 1990 年代以前の取引コストベースの研究は、複雑システムを扱う産業では組織統合が最も重要であるという主張を行っていた。たとえば Teece(1986)は「複雑なシステムは、設計要素間の依存関係が複雑であり、イノベーション活動（研究開発・製造販売等）間に強い依存関係がある。だから、たとえある企業が研究開発で優れたイノベーションを起こしたとしても、その企業が競争上有利になることは保証されない。むしろ製造販売で勝っている企業が、他の企業のイノベーションを活用して利益を上げてしまうかもしれない。」と主張していた。このような主張が行われたのは、初期の取引コストベースの研究は複雑なシステムはすべてインテグラルであるはずだという暗黙の仮定があったためであると思われる。

ところが、このような主張は、1990 年代に生まれた新しいイノベーションパターン、すなわちモジュラーアーキテクチャを基盤としたイノベーションが顕著になってくるに従って、修正が必要となった。

Teece は初め研究から 10 年後に「イノベーションのパターンによって組織間統合のパターンも変わるのではないか」というように主張を拡大した（Chesbrough and Teece, 1996）。あるイノベーションが、補完的なイノベーションを必要とし、補完的なイノベーションとの依存関係が強い場合（systemic innovation の場合）、組織間統合はあいかわらず重要である。しかし、補完的なイノベーション間が、産業標準規格で簡明に区分することが出来る場合、組織間統合は重要ではなくなり、各組織が独立してイノベーションを起こす自律的なイノベーション(autonomous innovation)が起こる。この場合、組織間統合は重要では無くなり、各専門分野をいかに上手く対処するのかが重要になる(Chesbrough and Teece, 1996)。

この研究(Chesbrough and Teece, 1996)は明らかに初期の研究(Teece, 1986)と異なり、アーキテクチャタイプが異なる場合、最適な組織間統合のあり方が異なることを指摘している。すなわち「コンポ

ーネント間が複雑に結合しているインテグラルアーキテクチャの場合、組織間統合が重要である。しかし、明確に標準規格によってコンポーネントが区分されるのであれば（すなわちモジュラーアーキテクチャであるのならば)、組織間統合は重要ではなくなる」という主張が行われたのである。

1990 年代初頭の研究と 1990 年代半ばの研究の共通点・相違点

まとめると、1990 年代初頭の研究と、1990 年代半ば以降のアーキテクチャ研究の間には、次の共通点・相違点が存在する(表 1)。

		1990年代初頭の研究	1990年代半ばの研究
共通点	対象としている人工物	複雑な人工物	複雑な人工物
	階層構造(モジュール化)	重視	重視
相違点	対象アーキテクチャ	インテグラル	モジュラー
	モジュール間の結合関係	複雑	簡明
	成功要因	組織間を統合すること	産業標準を主導すること
	既存企業が新規企業に打ち負かされる理由	既存組織が組織的認識限界により失敗するため	新規企業がネットワーク外部性により競争優位を構築するため

表 1　1990 年代初頭と半ばの研究の共通点・相違点

両研究で共通しているのは、複雑な人工物では階層化が行われるという点である。（これは準分解システムの①の特性である)。全ての設計要素間で依存性が均等に発生するわけではなく、設計要素間の依存性が高い部分（＝モジュール内）と、設計要素間の依存性が低い部分（＝モジュール間）に分別される。このモジュールが階層を構成する。

言い換えれば、複雑なシステムは、モジュール間の依存性という上位階層（アーキテクチャレベル）と、モジュール内の依存性という下位階層（コンポーネントレベル）で分離が生じる点が確認されたのである。

両研究の違いは「階層化されたアーキテクチャのうち、どこでイノベーションが起きることが競争構造を一変させるような重要な事態を引き起こすのであろうか」についての考察である。

1990 年代初頭の研究では、「アーキテクチャレベルのイノベーションのみが重要である」との見解を示していた。あるコンポーネント技術でイノベーションが起こったとしても、結局それは、依存関係のある他のコンポーネント技術なしに製品として実現する事は出来ない。だから、コンポーネントレベルのイノベーションよりもアーキテクチャレベルのイノベーションが重要であると考えたわけである。この背景には「複雑システムはインテグラルアーキテクチャである」との暗黙の前提

があった。

1990 年代初頭にアーキテクチャ研究で取りあげられたのは、半導体製造装置産業(Henderson and Clark, 1990)、ハードディスク産業(Christensen, 1992a, 1992b)、製薬産業(Henderson and Cockburn, 1994)であった。さらに、これらの研究の結果と主張が一致する研究として自動車産業(Clark and Fujimoto, 1991)があげられる。これらの産業は Ulrich(1995)の分類で言えば、設計要素間の対応関係が複雑なインテグラルアーキテクチャ製品の産業である。

これらの研究はコンポーネントレベルでのイノベーションに重きをおかず、アーキテクチャレベルのイノベーションが重要であることを繰り返し主張していた。そしてアーキテクチャレベルで技術革新がおきると、新規企業が既存企業に打ち勝つ可能性が高くなる点や、アーキテクチャレベルの技術革新をマネジメントするには組織の境界を越えたマネジメントが重要である点を強調していた。

ところが、1990 年半ば以降の研究では、準分解システムの②③の特徴を考慮し、アーキテクチャレベルの結合状態（すなわちモジュール間の結合状態）によっては、マネジメント上の重要事項が異なるとした。「いつもアーキテクチャレベルのイノベーションや組織間統合が重要であるとは言い切れない」と主張したのである。

確かに、モジュール間の依存性が強いインテグラルアーキテクチャの場合、組織間統合が重要である。しかし、もしもアーキテクチャがモジュラーアーキテクチャであったら、組織間統合が最も重要であるとは言えない。モジュラーアーキテクチャではモジュール間調整にそれほど大きな労力をかける必要はない。なぜならモジュール間に「標準規格」という「設計ルール」が設定されているために、標準さえ守れば誰でも部品を組み合わせて製品を作ることができる。このような特徴はデジタル産業ではほぼ共通して見ることが出来る。

そしてさらに重要な点は、両アーキテクチャでは「新規企業が既存企業に打ち勝つ」という現象に対する説明が異なる事である。この現象は、先に紹介したように 1990 年代のアメリカ産業のイノベーションパターンに頻繁に見られた特徴である（宮田, 2001）。

インテグラルアーキテクチャでこの現象が起こるのは、既存企業がアーキテクチャルイノベーションへの対応に失敗するからである。そして既存企業が失敗する理由は「組織的な認識力の限界」であり、場合によっては、「分かっていても行動する事が出来ないという問題」であった。しかし依然として組織間統合が重要であることには変わりがない。

ところが、モジュラーアーキテクチャでは、コンポーネントを提供している新規企業が強い競争

力を獲得することによって、既存企業に打ち勝つことが起こりえる。これは既存企業の失敗が原因ではない。そして、ここで重要な働きをするのが産業標準である。インテルやマイクロソフト、シスコのような新規企業は巨大なシステムの中の限られた部品（サブシステム）を提供する企業に過ぎなかったが、産業標準を味方につけることによって IBM 等の既存企業へと打ち勝っていった。一度、産業標準が広がれば、そこにネットワーク外部性が発生するため、その標準を握っている企業がますます強くなるという現象が起こる(Varian and Shapiro, 1999; Economides, 1996)。このためモジュラーアーキテクチャ下の産業では、たとえ既存企業が失敗しなくても、新規企業が産業の主役となることが出来るのである。

産業標準の重要性の指摘は、互換標準(Langlois and Robertson, 1992)、標準(Chesbrough and Teece, 1996)、見える化された設計ルール(Baldwin and Clark, 2000)など、様々な表現で各研究に見られるようになる。モジュラーアーキテクチャでは組織間統合にかわって産業標準（標準化）が重要な働きをするのであり、標準化プロセスの詳細な研究が 2000 年以降になって行われるようになった(Gawer, A. and Cusumano, M., 2002; Iansiti, M. and Levin, R., 2004; Winn, 2005; Chesbrough, Vanhaverbeke and West, 2006; Greenstein and Stango, 2007; 新宅・江藤, 2008; 小川, 2009; 立本・高梨, 2010)。

2.4 静的研究から動的研究へ(2000 年以降の研究)

2000 年以降、アーキテクチャ研究は静的研究から動的研究へとシフトしている。静的研究ではアーキテクチャ・タイプの違いが、イノベーションパターン、とくに組織間統合に影響を与えていることが分かった。組織間統合のあり方が異なるので、2 つのアーキテクチャ下ではイノベーションのマネジメントが異なる。インテグラルアーキテクチャでは組織間統合は重要な成功要因であった。しかしモジュラーアーキテクチャでは、組織間統合よりも産業標準を上手く扱うことが重要な経営手法である。だとすると、マネジメント上の重要な問いは、「アーキテクチャはどのような時にインテグラルなアーキテクチャになるのだろうか、また、どのような時にモジュラーアーキテクチャになるだろうか。」というアーキテクチャ変化の動的プロセスに向けられる。

設計進化プロセスの進化経路

設計進化プロセスについて、はじめて言及したのは、先に紹介した Simon(1962)である。しかし、その後、この考え方は長い間放置されていた。2000 年以降の動的プロセス研究でも、重要視されているとは言い難い。しかし第 3 節で述べるように、アーキテクチャを測定しようと考える際には、

設計進化プロセスが動的にどのように変化するのかを整理してモデル化する必要がある。

複雑な人工物は、複雑性を軽減する方向へと設計進化プロセスを経る。しかし、そのプロセスは単一の経路ではなく、潜在的にはいくつものバリエーション（選択肢）がある。一般に考えられているように、「複雑な人工物は問答無用にモジュラーアーキテクチャへ進化する」というのは、やや単純化しすぎた議論であり、本質を見落とす可能性がある。

重要なポイントは、モジュラーアーキテクチャへと進化する時ですら、さまざまな進化経路が潜在的に存在するという点である。設計進化は一本道ではないのである。そして、進化経路を詳細に検討することによって、「どのような」モジュラーアーキテクチャへと複雑な人工物が設計進化していくのかを知ることが出来る。設計進化が一本道でないのと同様に、設計進化によって実現されるモジュラーアーキテクチャも一つではないのである。

モジュラー化が進行したときに、実現可能なモジュラーアーキテクチャはいくつも存在する。そして自社に有利なモジュラーアーキテクチャがある一方で、自社に不利なモジュラーアーキテクチャが存在する。だから、進化経路に直接的・間接的に働きかけて、自社に有利なモジュラーアーキテクチャを実現する事が、マネジメント上重要である。

モジュラーアーキテクチャへの進化経路が複数ある事を、Baldwin and Clark(2000, pp.123-146)では「モジュラーオペレータ」として説明している。モジュラーオペレータとは、複雑性を軽減する際に、設計者がどのような設計簡素化方法を採用し、それがどのようなモジュール化を導くのかを整理した考え方である。たとえば Simon の時計職人の例にでてくるように、サブアッセンブリーという中間状態を作ることもモジュラーオペレータの 1 つである。Baldwin and Clark(2000)では 6 つの設計オペレータを提案している。モジュラーオペレータ毎に異なるモジュラーアーキテクチャが出現することになる。

しかしながら Baldwin and Clark(2000)では、詳細に検討したモジュラーオペレータは 6 つのモジュラー化オペレータのうちの一部にとどまる。また、どのモジュラーオペレータがどのような時に選択されるのかについて詳細に検討していない。このため、両氏の研究では一貫して単調なモジュラー化が行われるような印象を与える。また 6 つのモジュラーオペレータは、実際の設計行動を参考に提案したものであり、何が本質的なのかが詳細に検討されていない。

２つのモジュラー化の方向：カプセル化オペレータと共通化オペレータ

本稿では、オブジェクト指向設計方法を参考にして、より本質的であると考えられるモジュラー

オペレータを提案する。オブジェクト指向設計方法とは、設計要素間の関係性に着目してモジュール化をおこなう設計方法であり、ソフトウェアシステム設計では広く用いられている。

オブジェクト指向設計法をモジュラー化の視点から検討すると、本質的なモジュラー化オペレータは 2 つある事が分かる。１つめが「カプセル化オペレータ」であり、２つめが「共通化オペレータ」である。どちらもアーキテクチャをモジュラーアーキテクチャに導く、モジュラーオペレータである(図 2)。

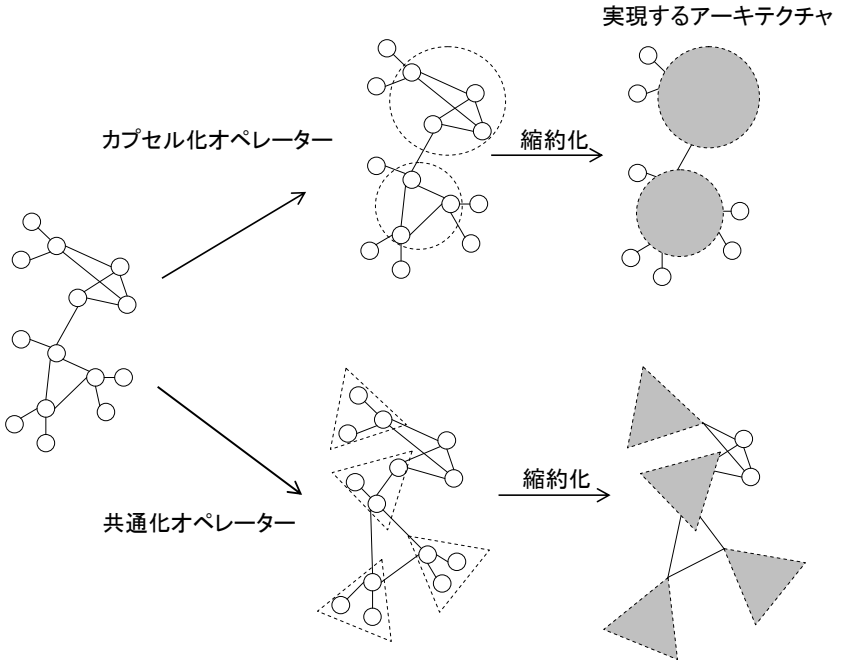

図 2 カプセル化オペレータと共通化オペレータ

カプセル化オペレータは、「has-a」関係と呼ばれる特性に着目してモジュール化を行うプロセスである。ある設計要素を考えたときに、その設計要素が他の設計要素に包含される場合は、「has-a」関係の 2 つの設計要素は結合している。「has-a」関係で結ばれる設計要素を集めていけば、Simon の時計職人の例で出たようなサブアッシータイプのモジュール化が行われる。カプセル化は情報隠蔽化とも呼ばれる。

共通化オペレータは、「is-a」関係という要素間結合を重視したモジュール化を行うプロセスである。「is-a」関係とは、あるモジュールとあるモジュールが、本質的に「同じ」ものであるのか、「異

なる」ものであるのかを検討した関係である。もしも「同じ」ものであるのであれば、新しいモジュールをつくるのではなく、同じモジュールを見本（基底）にして、若干の変更部分についてのみ修正を加えればよい。このような作業をオブジェクト指向設計法では継承とよぶ。継承とは、言い換えれば、モジュールの共通化の事である。

複雑システムの中で「is-a」関係を見つけ出していけば、共通モジュール（共通部品）が適用できる範囲が増え、モジュール化が進む。「is-a」関係で結ばれる設計要素を集めていけば、ある時計の内部に同じタイプのねじ（共通ねじ）を用いる、というような共通化（標準化）タイプのモジュール化が行われる。

モジュール化の研究で広く利用されるのはカプセル化オペレータである(青木, 1995)。一部の研究ではカプセル化がモジュール化のすべてあるとされることもあるが、それは正しくない。実際には「カプセル化オペレータ」と「共通化オペレータ」のように複数のモジュラーオペレータが存在する。どのモジュラーオペレータもモジュラー化を促進する。しかしモジュラーオペレータが異なると、異なるモジュラーアーキテクチャが実現する。最終的に実現されるモジュラーアーキテクチャに様々なバリエーションが生まれる。

モジュラー化：凝集性と分割性

次にモジュール粒度の問題について説明する。たとえば自動車は、数百点の部品で出来ているともいえるし、さらに細かいレベルの部品まで考えると数万点の部品で出来ているともいえる。なるべく大きなかたまりとして、設計要素の集まりを見た方がよいのだろうか、それとも、やや小さいかたまりで設計要素の集まりを見た方が良いのだろうか。

モジュール粒度の問題は、いいかえれば、どの程度まで設計要素をかたまりとして見なすのがよいのか、という問題である。この問題を解くには、モジュラー化が２つの側面（凝集性と分割性）をもっていることに着目する必要がある。モジュラー化とは「モジュール内の依存性が強まり」かつ「モジュール間の依存性が弱くなる」事であるので、「凝集性が高くなり、かつ、分割性が高くなる」という現象である。つまり、凝集性と分割性はモジュラー化が進行したときの表裏一体の現象である

凝集性：凝集性(agglomerativeness)とは「どのように設計要素が集合して依存性がつよくなるのか」という性質のことである。先述のモジュラーオペレータは、この凝集性をモデル化したものである。

本稿ではモジュラーオペレータとして「カプセル化オペレータ」と「共通化オペレータ」を説明した。モジュラーオペレータを使って設計要素をあるかたまり（凝集）することを縮約化とよぶ。縮約化はもしくはブロック化、モジュール化、階層化とも呼ばれる。

分割性：分割性(divisiveness)とは「全体システムがどのよう分割されるのか」という性質のことである。設計進化の過程で、複雑システムは、どのように分割されるのかについて複数の選択肢を持っている。この選択肢の中から、どの選択肢が選ばれるのかに基準を与えるものが、分割基準である。分割基準は、分割の目的に応じて複数提案されている。

室田(2004)はグラフ論の立場から、「モジュール内の依存性を最大化し、かつモジュール間の依存性が最小化させながら、各モジュールが階層構造を実現する」分割基準として DM 分解を紹介している。Newman(2006)は組織内のコミュニティ抽出を念頭に置きながら「モジュール内の依存性を最大化し、モジュール間の依存性を最小化する基準」としてモジュラリティ Q 基準（Modularity Q）を提案している。Baldwin and Clark(2000)は、経済的な視点から、「モジュールの集合体（すなわち全体システム）からの収益と、各モジュールへの投資額の比がもっとも大きくなる基準」として設計オプション(Design Options)基準を提案している。1 社内だけで分割を行うのではなく、産業全体でシステムを分割する場合には、標準化プロセスの研究ともあわせて考察する必要がある。

どのような分割基準を採用するかについては、上記のように複数の選択肢があり、どのような目的で分割を行うのかに大きく依存する。この点については、稿を改めて論じたいと思う。

縮約化の重要性：インテグラル化と凝集化の錯誤

アーキテクチャのタイプの研究では、インテグラルとモジュラーという二項対立によって分析が行われている。注意深く概念の操作化を行えば、このフレームワークは間違いではない。しかし、多くの研究では拙速な方法によって測定を行っているため、現象を正しく解釈することが難しい。一番多く行われる間違いは、「縮約化」というプロセスを無視する事である。

インテグラル化とは、モジュール間の結合状態が強まる事である。これと似て非なるものは凝集化で、モジュール内の結合状態が強まる事である。インテグラル化と凝集化は厳密に区別する必要がある。いいかえれば、モジュラー化（モジュール間の結合の簡明化）とモジュール化（モジュール内の結合の凝集化）を区別する必要があるのである。あるアーキテクチャがインテグラルなのかモジュラーであるのかは、モジュール化後に判明する。モジュール化という階層化が行われた後の、

モジュール間の結合状態が問題なのである。

整理すると、モジュールが成立して、かつモジュール間の依存性が強い状態がインテグラルな状態である。それに対して、モジュールが成立して、かつ、モジュール間が簡明な結合である状態がモジュラーな状態である。

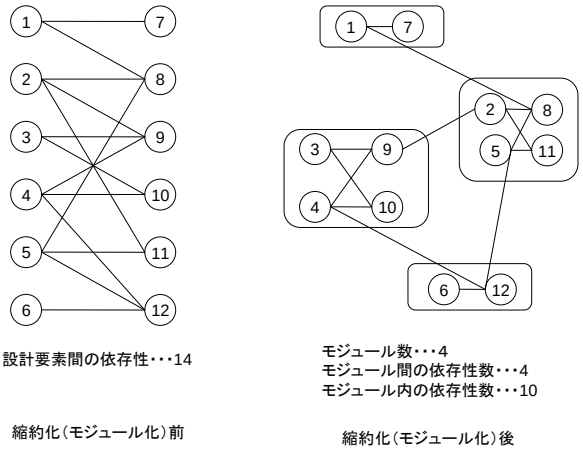

図 3 縮約化（モジュール化）の効果[2]

単純に設計要素間の結合状態の密度を測ることは、インテグラル化と凝集化を区別していない[3]。たとえば、図 3 左でしめされるネットワーク構造は、インテグラルであろうか、それともモジュラーであろうか。一見、複雑な結合構造であるので、インテグラルアーキテクチャであるように見えるが、縮約化を行った後の図(図 3 右)をみると、この結合構造は 4 つのモジュールで構成されていたことが分かる。この 4 つのモジュールに対して、モジュール間の依存性は 4 つに過ぎない。よって、図で示されているネットワークは依存関係が少ないモジュラーアーキテクチャである事が分かる。

この例示からわかるように縮約化というプロセスを行わない限り、アーキテクチャがインテグラルであるのかモジュラーであるのかを知ることは出来ない。「依存性が増加している」というだけで

[2] DM（Dulmage–Mendelsohn）分解の基準で縮約化を行った。DM 分解については室田(2004)を参照。
[3] このため、密度を使った研究では「ある特定の階層の設計要素にのみ密度指標を適応する」など注意深い処置が必要である。多くの研究ではこのような注意深い処置は行われていないため、問題が多い。また、単純な密度指標が十分な指標であるとはいえず、次善の方法として利用されている点にも注意が必要である。

は、モジュール内の依存性が増加しているのか、それとも、モジュール間の依存性が増加しているのかが区別できないからである。

インテグラル化と凝集化の錯誤は、縮約化（階層化）を考慮しなかったために起こる事態である。アーキテクチャが異なるとマネジメント上の問題が異なるため、この錯誤は実務上も重要な問題を引き起こす。この問題点は、かなり早い時期に青島・武石(2000)や藤本(2000)で指摘されている。何らかの形で縮約化を行った後のデータを評価することが重要であり、これを行っていない場合、重大な錯誤が発生する可能性が有る。

3 節　今後の研究の方向性：アーキテクチャ研究の方向性

動的プロセス研究のためのフレームワーク

2.4 項ではアーキテクチャの動的プロセスを研究する際に必要な、次の３つキーコンセプトについて説明した。

第 1 にモジュラーオペレータについて検討した。モジュラーオペレータ毎に異なるモジュラーアーキテクチャが実現する。どのモジュラーオペレータがどのような影響をもたらすのかを検証することが重要な点となる。モジュラーオペレータは複数存在し（例えば「カプセル化オペレータ」と「共通化オペレータ」)、潜在的なアーキテクチャの選択肢を複数作る。

第 2 に縮約化の重要性について述べた。モジュラー化を検証するためには、縮約化が必須のプロセスである。モジュラー化はモジュール間の依存性の状態を示したものであり、モジュール内の依存性の状態ではない。よって縮約化によってモジュールが生成された後でなければ、モジュラー化の状態を検討することはできない。技術経営にとって、モジュール化ではなく、モジュラー化が重要な特性である。

第 3 に分割基準の問題について説明した。モジュラー化は凝集性と分割性が同時に高まることである。凝集性はモジュラーオペレータによって実現される。分割性についてはいくつかの分割基準が提案されており、目的ごとに異なる。凝集性と分割性をあわせて考慮することで分割基準の問題を解くことが出来る。

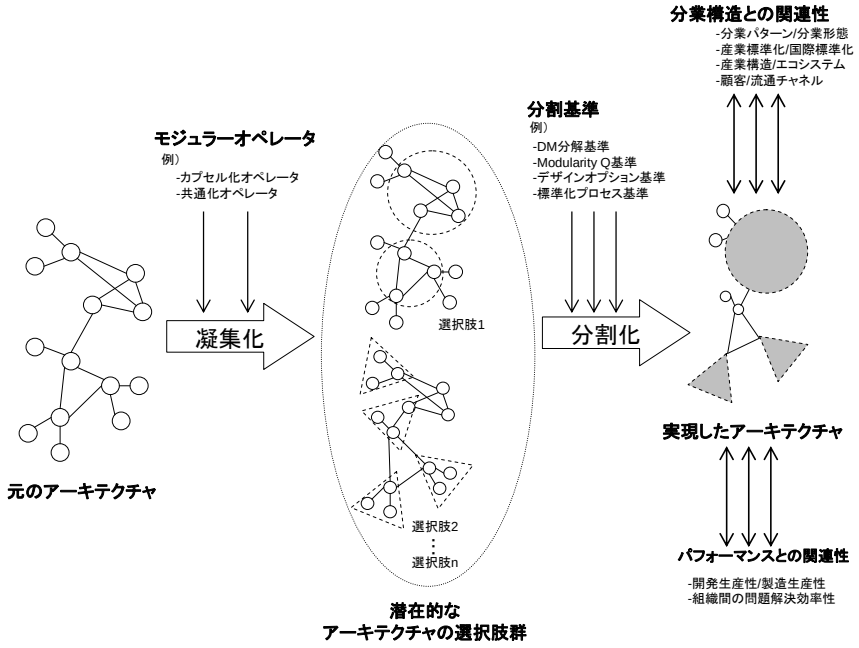

図 4 アーキテクチャの動的研究のフレームワーク

上述の３つのキーコンセプトを考慮したアーキテクチャの動的研究のフレームワークを示す（図4)。研究は大きく２つの領域に分かれる。１つめは狭い意味の動的プロセス研究であり、凝集化と分割化によって、複数存在する選択肢群からどのようにアーキテクチャが実現するのかが主なテーマになる。すなわち「アーキテクチャの実現プロセス」である。凝集化、分割化ともにいまだ十分に研究されているとは言い難い。この点について複雑ネットワークや社会ネットワークの研究が大きく躍進しており(Watts, 2003; Newman, 2003)、今後アーキテクチャ分野への適応が期待される。狭い意味の動的プロセス研究は、複雑な人工物がもつ特性それ自身について集中している。

2 つめは広い意味の動的プロセス研究である。「アーキテクチャが実現するプロセス」と外部指標がどのような関係にあるのかという点である。この研究は、人工物の設計進化と産業進化の関連性を主眼としている。外部指標とは主に分業構造との関連性、さらにパフォーマンスとの関連性である。技術経営の観点からは、この広い意味の動的プロセス研究が重要である。外部指標は分業構造とパフォーマンス指標の２つに区分できる。

静的研究からすでに特定のアーキテクチャと特定の分業構造の間には適合性があることが分かっ

ている。しかし、静的研究はすでに実現したアーキテクチャのタイプと、すでに存在する分業構造の対比を行っているに過ぎない。アーキテクチャが実現するプロセスで、どのような戦略的な選択肢が存在するのか、そしてそのような選択肢をとった場合どのような影響を分業構造とパフォーマンスに与えるのかについては、ほとんど検討されていない。

既存研究ではアーキテクチャに選択肢が無数に存在することを前提とした研究はほとんど存在しない。たとえば「設計進化プロセスに対して直接的・間接的に影響する主要要因は何であるのか」、「どのようにすれば設計進化プロセスを戦略的に活用する事が出来るのだろうか」というような点は、現時点では明らかにされているとは言い難い状態である。複数の選択肢（潜在的なアーキテクチャの選択肢）から、「どのようなアーキテクチャを選択するべきなのか」を検討をすることによって、企業戦略論の立場から複雑な人工物について扱えるようになるだろう。

2000 年以降の動的プロセス研究：設計進化プロセスと標準化プロセス

2000 年以降、アーキテクチャ研究は動的プロセスに関心が集まっていることを述べた。特にインテグラルアーキテクチャがモジュラーアーキテクチャに移行する時には、大きな産業構造変化やイノベーションパターン変化が生じることが知られており、この点に注目が集まっている。

たとえばコンピュータ産業は、1960 年代のメインフレームの時代にはインテグラルアーキテクチャであったが、その後は、モジュラー化の設計進化プロセスをあゆみ、1990 年代のパソコンはモジュラーアーキテクチャの代表的な製品となった。これに対応するように、垂直統合型の産業構造は、水平分業型（垂直断裂型）の産業構造へと急激に変化していった。その象徴が、総合システム企業であった IBM の凋落と、コンポーネント企業であったインテル、マイクロソフトの台頭である。この傾向は、デジタル産業では共通して見られ、他産業へも影響を拡大している。従来デジタル化とは無縁であると考えられていた自動車産業ですら、組込システムを大量に取り入れた結果、デジタル化の影響を受けていると言われている(徳田, 2008; 徳田・立本・小川, forthcoming)。

アーキテクチャがインテグラルからモジュラーに移行する際には、２つの動的プロセスを経ている。１つは人工物設計の立場から見たときに、複雑性軽減を目的にした設計進化プロセスである。先述の動的プロセス研究のフレームワークで紹介した「モジュラーオペレータ」「分割基準」などが主要な分析対象となるだろう。静的研究ではモジュラーアーキテクチャを単一のものと考えていたが、動的研究ではモジュラーアーキテクチャのバリエーションに関心が高まっている。また複雑システムが階層性を持つことについて再び焦点が当たっている。動的プロセスでは、階層毎にことな

るアーキテクチャ上の変化が観察される可能性がある。ある階層ではモジュラー化が進んでいるが、ある階層ではインテグラル化が進んでいるという事態が起こりえる。階層間の動的プロセスの関連性も重要な研究対象であろう。

2 つめは、あるインターフェースを組織間で共有しようとする標準化プロセスである。プラットフォームビジネス研究(Gawer and Cusumano, 2002)や産業エコシステム研究(Iansiti and Levin, 2004)は、標準化を活用した企業戦略を研究している。標準化プロセスは、大きく言えば設計進化プロセスの一部であるが、制度の影響（独禁法や標準化政策等）を強く受けるプロセスであるため、設計進化プロセスとは別に研究されている（小川, 2009; 立本・新宅・小川, 2010; 立本・高梨, forthcoming）。1980 年代に行われた各国の標準化政策の転換（独禁法の緩和、地域標準の重視、WTO/TBT 条約）が、1990 年代の標準化プロセスに大きな影響を与えており、標準化を活用した企業戦略の重要性を高めている。

２つの動的プロセスの研究が、2000 年以降のアーキテクチャの動的研究の主流になっている。設計進化プロセスと標準化プロセスは互いに影響し合うプロセスであり、この 2 つのプロセスを十分に理解し、戦略的に活用する事がモジュラーアーキテクチャ下での成功要因であると考えられている。動的プロセスの視点から、今後さらなるアーキテクチャ研究が期待されている。

参考文献

Alexander, C. (1964) *Notes on the Synthesis of Form*, Harvard University Press.

Asanuma, B. (1989) Manufacturer-supplier Relationships in Japan and the Concept of Relation-specific Skill, *Journal of the Japanese and International Economies*, Vol.3, No.1, pp.1-30.

Burt, R.S. (1976) Positions in networks, *Social Forces*, Vol. 55, pp.91-122

Baldwin, K. Y. and Clark, K. B. (2000) *Design rules: The power of modularity*, MIT Press. （邦訳、カーリス・Y・ボールドウイン、キム・B・クラーク（2004）『デザイン・ルール』安藤晴彦訳、東洋経済新報社.）

Borgatti, S.P., Everett, M.,G.. and Freeman, L., C. (2002) *Ucinet 6 for Windows: Software for Social Network Analysis*, Analytic Technologies.

Henderson, R.,M. and Clark, K., B. (1990) Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms, *Administrative Science Quarterly*, Vol.35,

pp.9-30.

Clark, K. B. (1985) The interaction of design hierarchies and market concepts in the technological evolution, *Research Policy*, Vol. 14, pp.235-251.

Clark, K., B. (1989) Project scope and project performance: The effect of parts strategy and supplier involvement on product development, *Management Science*, Vol.35, No.10, pp.1247-1263.

Christensen, C., M. (1992a) Exploring the Limits of the Technology S-Curve, Part I: component Technologies, *Production and Operations Management*, Vol.1, pp.334-357.

Christensen, C., M. (1992b) Exploring the Limits of the Technology S-curve, Part 2: Architectural Technologies, *Production and Operations Management*, Vol. 1, pp.358-66.

Christensen, C. M. (1997) *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*, Harvard Business Press.

Chesbrough, H., W. and Teece, D., J.(1996) Organizing for Innovation: When Is Virtual Virtuous?, *Harvard Business Review*, 1996 January-February, pp.65-73.

Chesbrough, H., Vanhaverbeke,W. and West, J. (2006) *Open Innovation: Researching a New Paradigm*, Oxford Univ. Press.

Dyer, J. H. and Nobeoka, K.(2000) Creating and managing a high-performance knowledge-sharing network: the Toyota case, *Strategic. Management Journal*, Vol.21, pp.345-367.

Economides, N. (1996) The Economics of Networks, *International Journal of Industrial Organization*, Vol.14, pp.673-699.

Gawer, A. and Cusumano, M. (2002) *Platform Leadership: How Intel, Microsoft, and. Cisco Drive Industry Innovation*, Harvard Business Press.

Iansiti, M. and Levin, R. (2004), *The Keystone Advantage*, Harvard University Press .

Langlois, R., N. and Robertson, P., L. (1992) Networks and innovation in a modular system: Lessons from the microcomputer and stereo component industries, *Research Policy*, Vol. 21, pp. 297-313.

MacCormack, A., Runsnak, J. and Baldwin, C., Y. (2006) Exploring the structure of complex software designs: An empirical study of open software and proprietary code, *Management Science*, Vol.52, No.7, pp.1015-1030.

Newman, M., E.(2003) The structure and function of complex networks, SIAM Review, Vol.45, pp.167-256.

Newman, M., E., J. and Girvan, M. (2004) Finding and evaluating community structure in networks, *Physical Review E*, Vol.69, No.12, pp.1-16.

Robertson, P., L. and Langlois, R., N. (1995) Innovation, network and vertical integration, *Research Policy*, Vol.24, pp.543-562.

Schwartz, J.E. (1977) An Examination of Concor and Related Methods for Blocking Sociometric Data, *Sociological Methodology*, Vol.8, pp.255-282.

Shapiro, C. and Varian, H. R. (1999) *Information Rules*, Harvard Business School Press.

Simon, H., A. (1962) The Architecture of Complexity, *Proceedings of the American Philosophical Society*, Vol. 106, No. 6. (Dec. 12, 1962), pp.467-482.

Greenstein, S. and Stango, V. (2007) *Standards and Public Policy*, Cambridge Univ. Press.

Takeishi, A. (2002) Knowledge Partitioning in the Interfirm Division of Labor: The Case of Automotive Product Development, *Organization Science*, Vol. 13, No.3, pp.321-338.

Teece, D., J. (1986) Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy, *Research Policy*, Vol.15, pp.285-305.

Ulrich, K., T.（1995）Product Architecture in the manufacturing Firm, *Research Policy*, Vol. 24, pp.419-440.

Wiener, N. (1948) Cybernetics, MIT Press. (邦訳　『サイバネティクス第二版』(1961) 岩波書店)

Williamson, E. O. (1979) Transaction-Cost Economies: The Governance of Contractual Relations, *The Journal of Law and Economics*, Vol.22, pp.548-577.

Winn, J., K. (2005) US and EU Regulatory Competition in ICT Standardization Law and Policy, *IEEE. SIIT2005 Proceedings*, pp.281-291.

Watts, D.,J. (2003) Six Degrees –The science of connected age, W.W. Norton & Company, New York.(邦訳 辻竜平・友知政樹『スモールワールドネットワーク－世界を知るための新科学的思考法』阪急コミュニケーションズ(2004))

青木昌彦(1995)『経済システムの進化と多元性：比較制度分析序説』東洋経済出版社.

青島矢一・武石彰(2001) 「アーキテクチャという考え方」所収 藤本・武石・青島(2001) 第 2 章.

小川紘一(2009) 『国際標準化と事業戦略―日本型イノベーションとしての標準化ビジネスモデル』

白桃書房.

奥野 正寛・瀧澤 弘和・渡邊 泰典(2006)「人工物の複雑化と製品アーキテクチャ」*MMRC Discussion Paper*, No.81.

金光淳(2003) 『社会ネットワーク分析の基礎：社会的関係資本論にむけて』勁草書房.

新宅純二郎・江藤学（2008）『コンセンサス標準戦略 –事業活用のすべて』日本経済新聞出版社.

立本博文・小川紘一・新宅純二郎(2010) 「オープン・イノベーションとプラットフォーム・ビジネス」『研究技術計画』Vol. 25, No. 1. (forthcoming)

立本博文・高梨千賀子(2010)「標準規格をめぐる競争戦略-コンセンサス標準の確立と利益獲得を目指して」『日本経営システム学会誌』Vol.26, No. 2, pp. 67-81.

徳田昭雄(2008)『自動車のエレクトロニクス化と標準化―転換期に立つ電子制御システム市場』晃洋書房.

徳田昭雄・立本博文・小川紘一編(forthcoming)『自動車組込みシステムの開発と標準化 : 欧州オープン・イノベーションの実態』晃洋書房.

藤本隆宏(2001)「アーキテクチャの産業論」所収 藤本・武石・青島(2001) 第 1 章.

藤本隆宏(2009) 「複雑化する人工物の設計・利用に関する補完的アプローチ」『横幹』第 3 巻,pp.52-59.

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.

宮田由紀夫(2001) 『アメリカの産業政策－論争と実践』八千代出版.

室田一雄(2004) 混合行列の正準形と階層構造『数学セミナー』Vol. 513, pp.38-43.

安田雪(2001) 『実践ネットワーク分析 : 関係を解く理論と技法』新曜社.

鈴木努(2009) 『ネットワーク分析』共立出版.