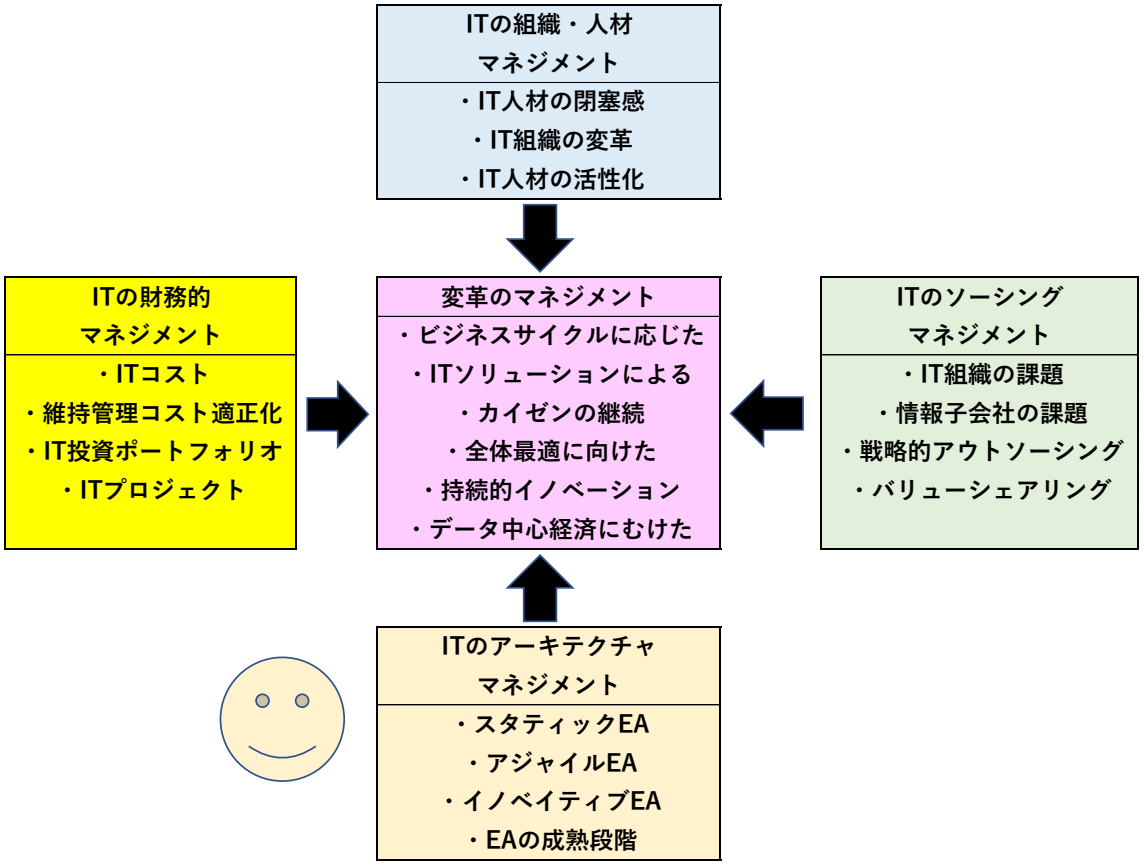
IT による変革の方法論集

ある IT コンサルタントのツールボックス

IT マネジメント編

IT のアーキテクチャマネジメント



日本 IT ガバナンス協会　理事

博士（商学）　淀川　高喜

yodokouki@ktd.biglobe.ne.jp

内容

EA の概念の進化

不確実性の時代で重要性を増す EA

EA（エンタープライズアーキテクチャ）は、ジョン・ザックマンが 1987 年に情報システムの全体構造を示す方法として提唱したものを下敷きとして、1999 年に米国連邦政府の業務と情報システムの全体構造を示す方法に拡張された。

・BA（ビジネス・アーキテクチャ） 業務（プロセス）の全体構造
・DA（データ・アーキテクチャ） データの全体構造
・AA（アプリケーション・アーキテクチャ） 業務システムの全体構造
・TA（テクノロジー・アーキテクチャ） IT 基盤の全体構造

を、整合性を持たせて記述した青写真を描いて、その理想形にむけて実際の業務と情報システムを整備することを目指した。

EA は、ビジネスと IT 資産の整合性を確保し、IT 活用の全体最適化を図るための拠り所となる全体設計図である。ビジネスの不確実性、技術や製品の不確実性、IT 運営の不確実性が高まっている状況では、変化に即時に対応して、整合性を保ちつつ IT 資産を変更する必要がある。そのために EA を規定し、目指すべき理想形（To-Be）と現実の姿（As-IS）の両面から、業務と情報システムの姿を関係者間で共通認識しておくことが有効である。

BA-DA-AA-TA の順番であるが、プロセス（ビジネス）とデータは論理的モデル、アプリと IT 基盤はその実装形態である。一方、プロセスとアプリは情報システムの実行内容、データと IT 基盤は実行のためのリソースである。したがって、BA-DA-AA-TA は設計時の全体構造、BA-AA-DA-TA は実行時の全体構造といえる。

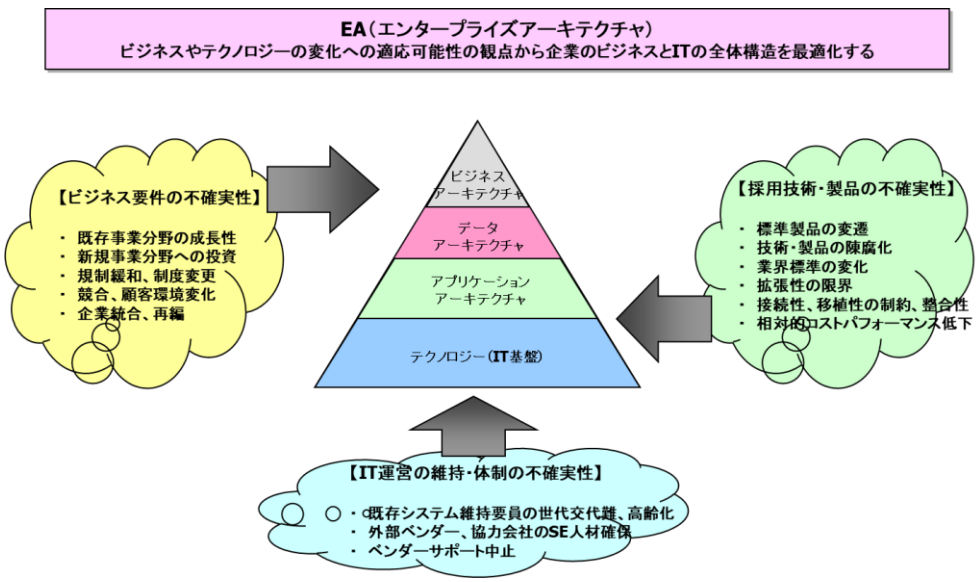


図 1 不確実性の時代で重要性を増す EA 筆者作成

EA の始まりは IT 基盤の標準化から

標準的なアーキテクチャの規定は、IT 基盤の分野から始まった。

アプリケーションごとに別々の IT 基盤構成や製品が採用されることが無いように、IT 基盤の標準形を規定して、IT 基盤全体の品質や性能を確保し、システム構築の生産性と品質を高め、技術や製品の世代交代に速やかに対応できるようにすることが目的である。

IT 基盤標準に規定する内容は以下の通りである。

・IT 基盤整備の基本方針、基盤基本構造、技術・製品の選択基準、IT 技術進歩のロードマップを全体アーキテクチャとして記述する。

・それをもとに、IT 基盤の構成要素ごとに個別アーキテクチャ標準を記述する。処理方式パターン、システム構築ガイドライン、推奨製品、設計書ひな形、開発標準などが含まれる。

・個別アーキテクチャを参照して、実際の IT 基盤設計開発を行う。開発経験に基づくノウハウはアーキテクチャにフィードバックされ反映される。

これは 4 階層で示される現在の EA の中に、テクノロジー・アーキテクチャとして継承されている。

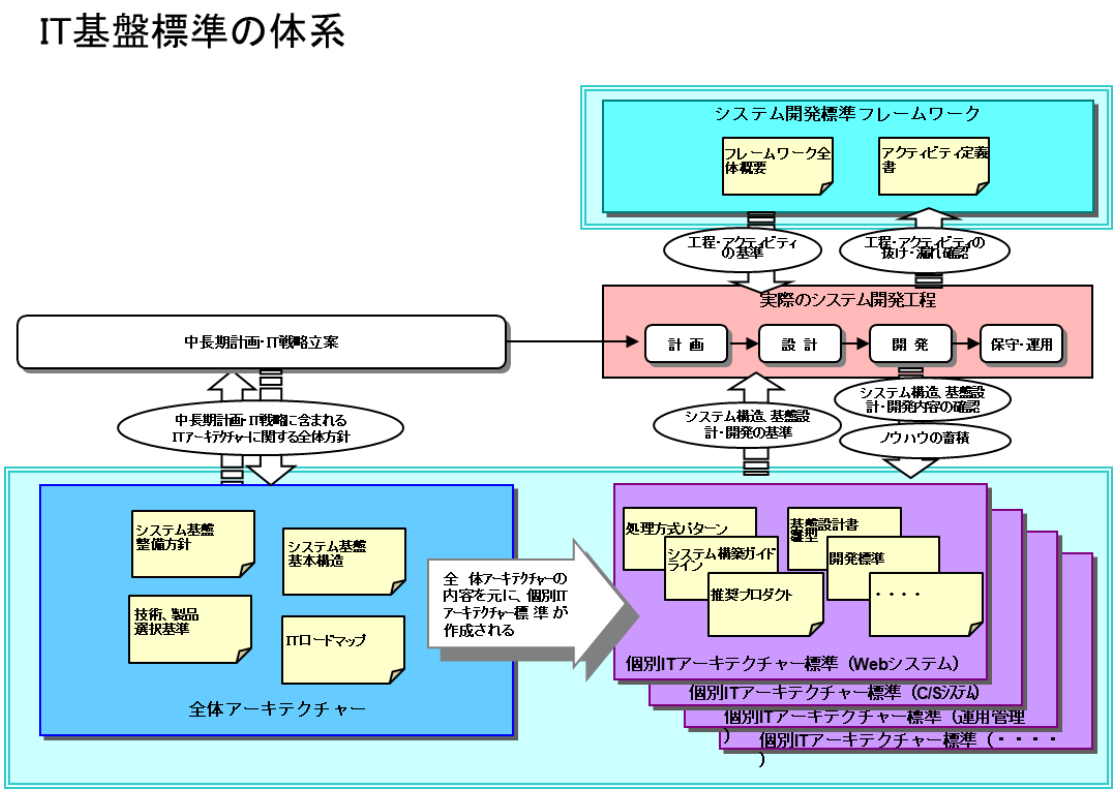


図 2　IT 基盤の標準化　NRI

EA は標準化アプローチの集約

EA の目的には、ビジネス変革のシナリオに応じて、経営戦略と IT 資産を整合性を保って構築するという経営視点からの文脈と、システムの品質や生産性を確保するという現場視点の文脈がある。このためには、IT 基盤だけでなく、業務、データ、アプリケーションも含めた IT 資産全体を EA の対象に含める必要がある。

EA を実現する方法は、4つの階層ごとに用意された方法論の組み合わせである。

たとえば、

・標準化された IT 基盤を生み出すためには IT 基盤標準の策定手法

・標準部品化されたアプリケーションを構築するには ERP（統合業務パッケージ）

・全社で一元管理された大福帳データベースを構築するには DOA（データ中心アプローチ）によるデータモデル設計

・標準業務プロセスのモデリングには BPM（ビジネスプロセスマネジメント）

が用いられる。

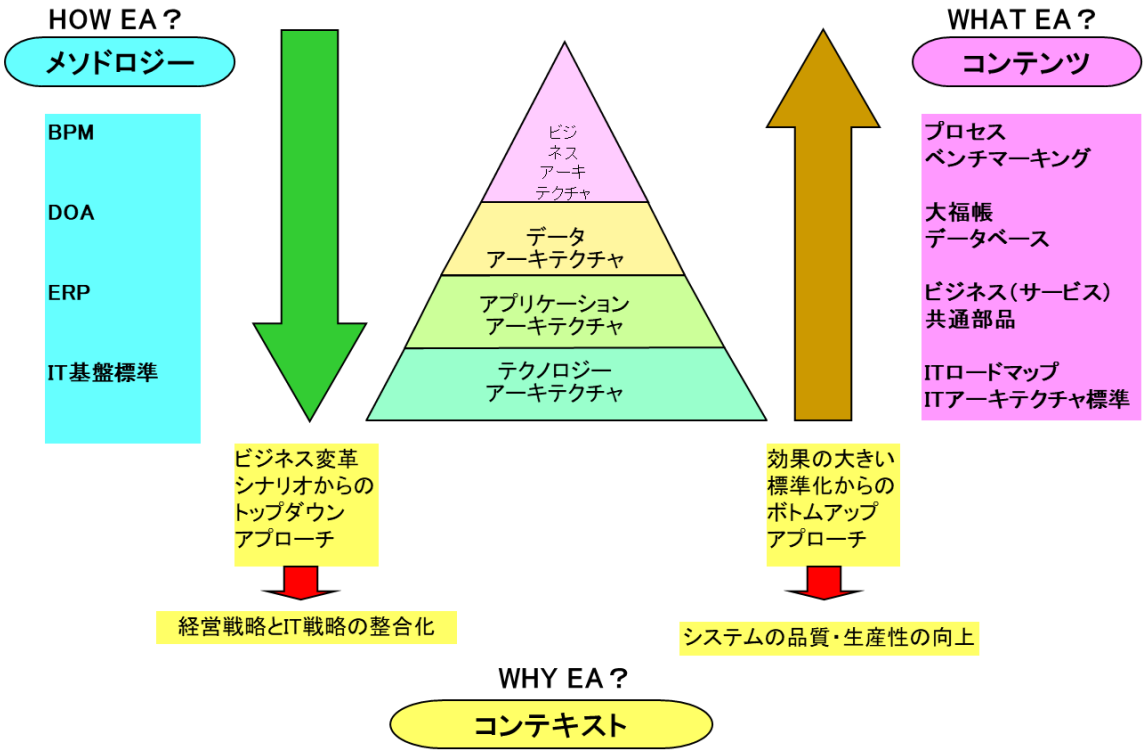


図 3　EA は標準化アプローチの集約　筆者作成

4つのITアーキテクチャ戦略

マサチューセッツ工科大学のジニー・ロス教授(2002)は,「個別最適型」,「IT基盤標準型」,「プロセス・データ統合型」,「共通部品型」という4つの段階をへて,企業のシステムの全体構造は成熟度を高めていくと提唱している。この成熟の段階を、事業のライフサイクルと照らし合わせてみると以下のようになる。

・起業から成長の段階において、成長に間に合うように次々と個別にシステムを追加していくと、「個別最適型」の業務システムの寄せ集めになってしまう。

・企業が成長から成熟の段階に移るにつれて、個別最適の集まりでは、システムを維持管理・運用していく上で非効率であるため、各業務システムを乗せている土台の部分であるハードウェアやネットワークなどの IT 基盤を、自社で定めた標準にそった技術や製品で構成し共通化を図るという「IT 基盤標準型」になる。

・成熟から統合に至る段階では、IT 基盤だけでなく全社の業務プロセスやデータを統合し全体最適化を図る「プロセス・データ統合型」の「ワン IT」が必要になる。

・事業の統合をより柔軟に行うために、あるいは分化した事業を自律性を確保しながら連合させるためには、共有や再利用が可能な部品として全社の業務機能を構成し、その組み合わせで各事業の業務プロセスが実現できる「共通部品型」の「リユース IT」 が求められる。

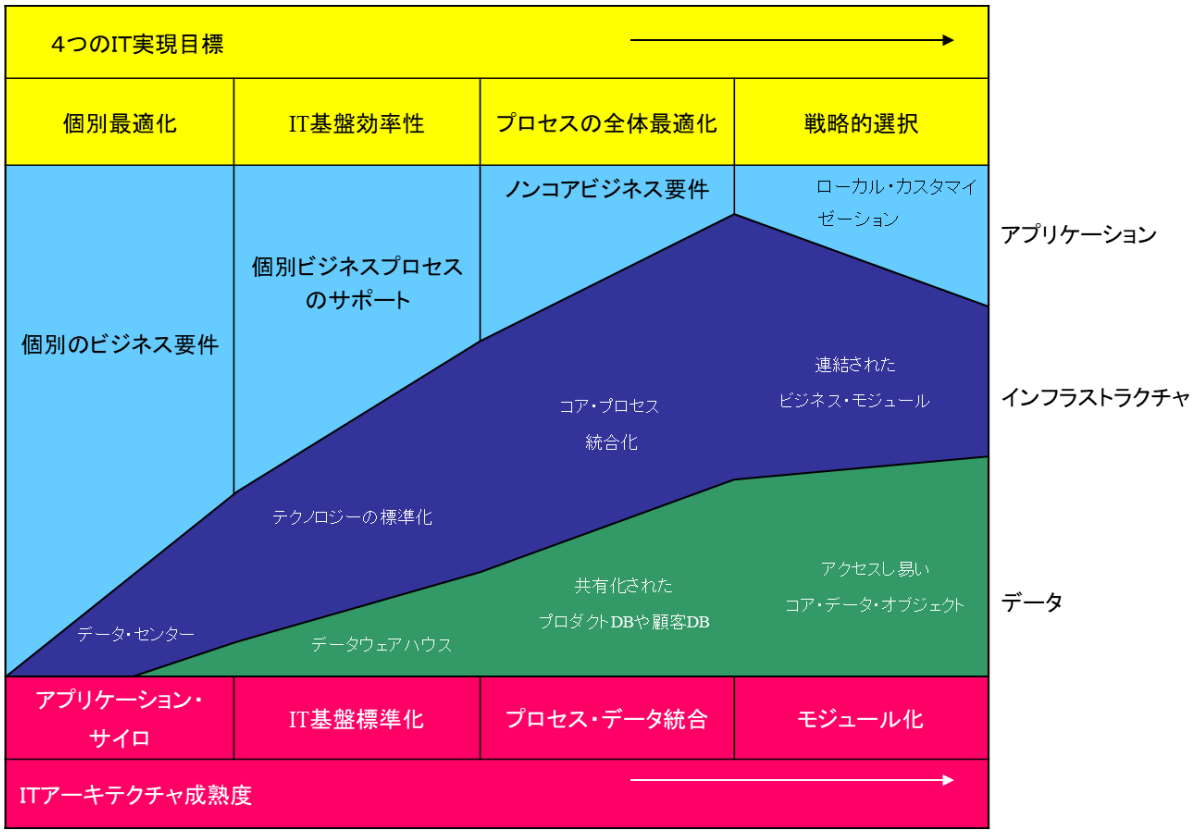


図 4　4つのＩＴアーキテクチャ戦略　ロス 2002

ERP を用いたプロセス・データ統合の実現

ERPは、標準化されたプロセスモデルとデータモデルを下敷きにして構築された統合業務パッケージである。ERPを適用して全社システムを構築すれば、プロセス・データ統合型のアーキテクチャが実現できる。

ERP製品は、絶えず機能拡張が行われており、従来からの業務処理的機能に加えて、OLAPが得意とするような、多次元データベースによる情報系システム機能やEIS（経営者情報システム）機能を中に取り込んだり、グループウェアが得意とするような、稟議決裁処理に代表されるワークフロー機能を追加したりしている。このため、ERPをオールインワンの全社システム実現手段とすることも可能になってきた。

しかし、一方で、それぞれの分野を得意技とする専用ソフトウェアをERPと組み合わせて使うことにも利点がある。

・OLAP, SFA, ワークフローといった各分野の専門ソフトウェアは、餅は餅屋であり、ERPが品揃えの一部として持っている同等の機能に比べて、機能が充実している。

・各分野でのマーケットシェアが高く、デファクト・スタンダード化している専門ソフトウェアを使うほうが、ベンダーのサポートが充実しており、最新技術の反映や、新機能の拡張スピードも速い。

・ERPベンダー自身、自社製品の機能として一通りは用意するものの、それ以上の要求に応えるため、有力な専門ソフトウェアのベンダーとは相互接続できるような連携を組んでいる。

・ERPが行う基幹業務処理と、OLAPが行う自由度の高い情報系機能や、SFAが行うフロント業務処理とは、システムの信頼性や性能要求や機能拡張頻度などの要件が異なり、物理的にもシステムの器を分けた方が、お互いに悪影響を与えない。あえて、ひとつのシステム、ひとつのデータベースとする必要性は少ない。

また、ERPを用いずに、自社独自システムの構築を選択すべき部分もある。

ひとつは、その業務が自社の差別化の源泉であったり、自社のアイデンティティに関わるものであったりして、どうしても、標準的な機能で済ませるのではなく独自性を主張したい部分である。

もうひとつは、ERP製品で提供される業務機能のレベルでは、自社の顧客サービスで要求される不可欠な業務レベルを達成できる見通しが立たない部分である。

高い顧客サービスレベルが要求される受注システムや、高度な統合ロジスティクスコントロールや、自社の独自の生産技術を組み入れた生産プロセス管理などがこれにあたる。

また、戦略性が高いわけではないが、取引先の都合もあって複雑な取引形態がどうしても標準化できない場合には、販売管理業務（契約、受注、出荷、納品、請求、売掛、入金、返品、リベート、販促費）のような社内に閉じない業務も、ERPにのせられない。

この場合でも、極力、「つくらずつくる」の方針で、ERPのデータモデルやシステム部品構成を参照して、全体としては統合したプロセスとデータが実現できるように設計すべきである。

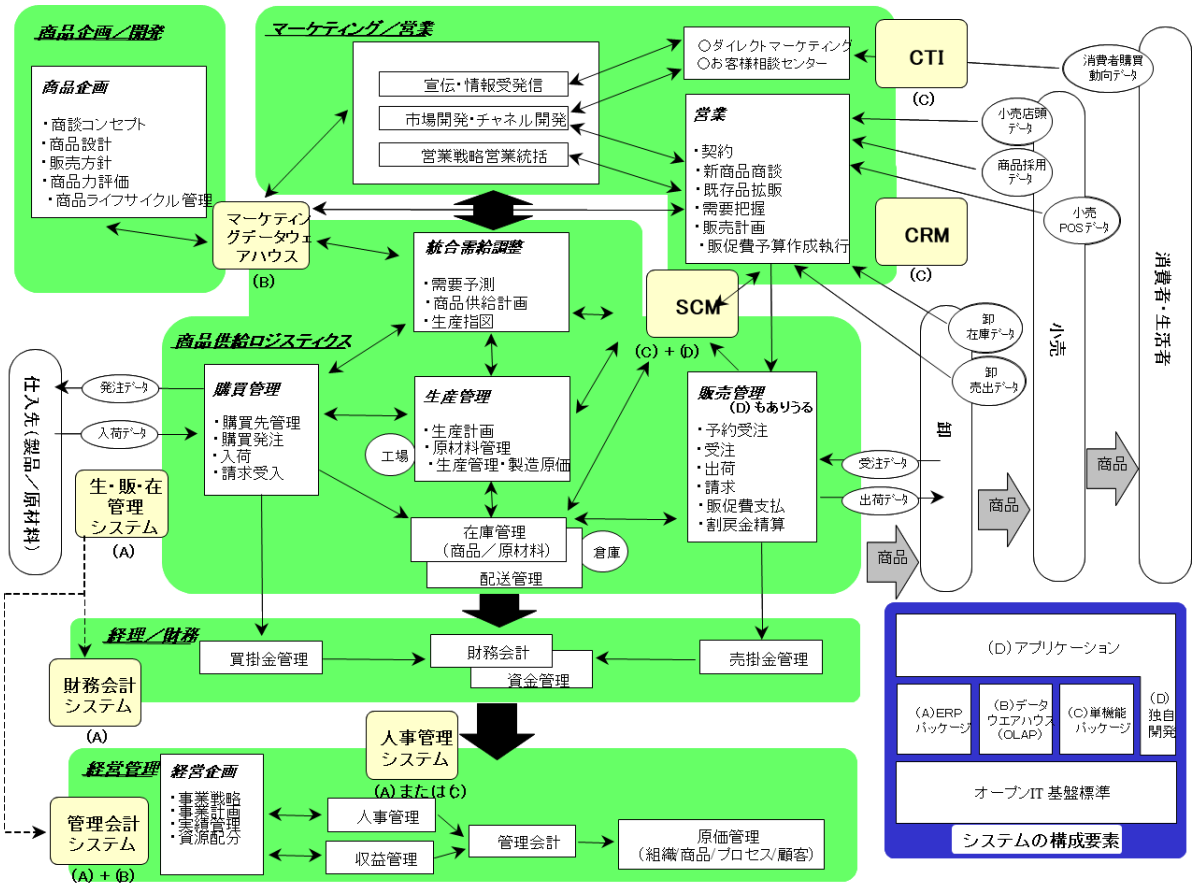


図 5　ERP を用いたプロセス・データ統合の実現　筆者作成

変化適応力のある EA への進化

先が読める時代においては、明確な経営戦略を実現するための最適なビジネスアーキテクチャを描き、それを効率よくサポートできるアプリケーションと IT 基盤のアーキテクチャを構築するというスタティックな EA で良かった。グループ全体最適を目指してプロセス・データ統合型の EA を構築する場合がこれである。

しかし、経営環境が不確実な時代においては、変化する経営戦略の選択肢に対応して、複数のビジネスアーキテクチャを想定し、アプリケーションの共通部品を組み合わせて変化に適応できるようなダイナミックな EA が必要になる。これが疎結合した共通部品の集合体としてモジュール型の EA を構築する理由である。

共通部品は、リユース（再利用）やリフォーム（改訂）が容易なモジュールとして設計し、自社ですべてを独自開発せずに、外部で利用可能な標準部品があればそれを組み合わせて使うようにする。情報システム資産を所有から利用に転換することによって、IT 投資を変動費化すれば、企業は変化にスピーディに適応できるようになる。このための方法論が SOA(サービス指向アーキテクチャ)である。

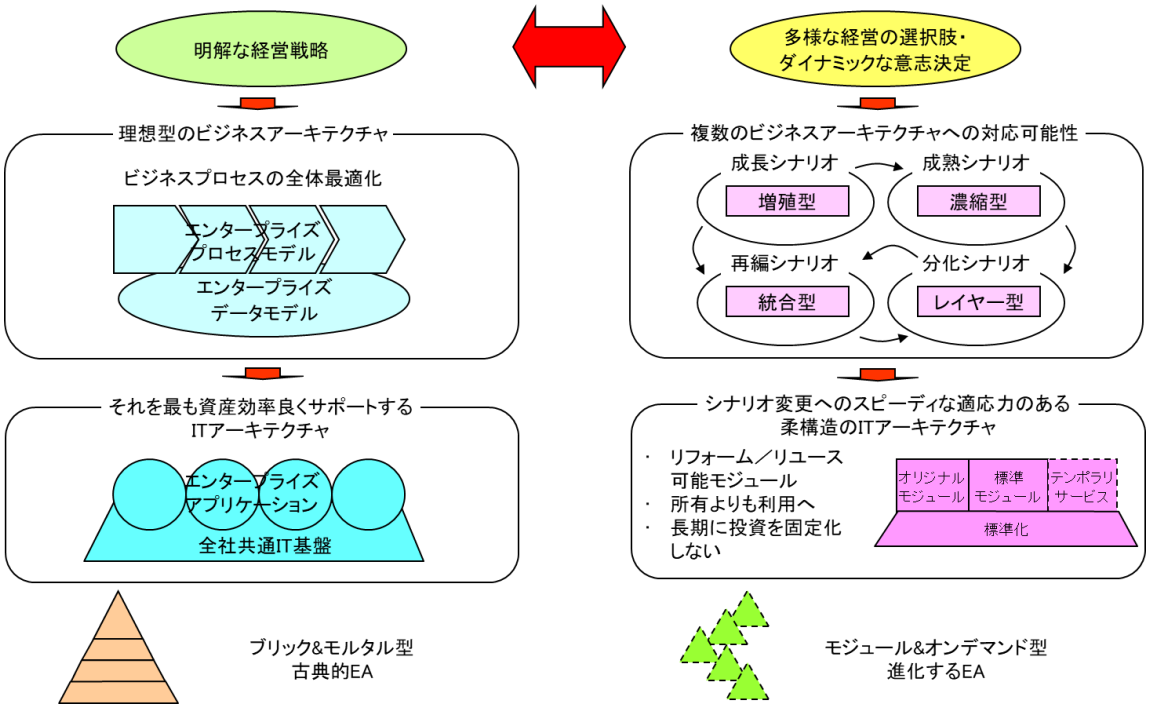


図 6　変化適応力のある EA への進化　筆者作成

アジャイルな企業のアーキテクチャとは

SOA（サービス指向のアーキテクチャ）とは、システムの構造だけを指すのではなく、その前提となるビジネスの構造も含めての全体構造である。

① 人的資産レイヤー

まず、企業の組織構造自体を、事業部門別などの縦割り組織と、その組織の間をつないで縦割り組織を超えたサービスを提供する部門横断組織からなる、連邦型に変える必要がある。そして、縦割り組織の自律的な活動と、部門横断組織の共通サービス提供活動とが、うまく連携することによって、臨機応変の柔軟な協働が実現する。

② プロセス資産レイヤー

次に、各組織が持っている知恵、情報、仕事のやりかたなどの暗黙知の中で、組織間で共有し、再利用する価値があるものについて、ビジネスルール、情報の意味の共通定義、それらを利用するためのサービス規約、ビジネス実行手順といった目に見える形にする必要がある。こうした共有化された形式知を使って各組織のビジネスプロセスを組み立て、組織を超えた知恵、情報、仕事のやりかたの再利用をすすめる。

③ システム資産レイヤー

さらに、形式知したビジネスルールや情報をシステムに埋め込んで、システムとしてのサービス・コンポーネントを作る。サービス規約をシステムに埋め込んでサービス・リポジトリを作る。ビジネスプロセスの流れを制御するビジネスプロセス・マネジメントシステム（BPM）を作る。各サービスをエンタプライズ・サービスバス（高速な社内ネットワーク）でつなぐ。こうして、サービス指向のシステムが出来上がる。

④ 物理的資産レイヤー

システムをサービスの集合体として構成してあれば、サービスのうちで、自社で保有しなくとも良いものは、外部のサービスを利用するように切り替えることが容易である。これによって自社を身軽にし、自社のリソースを本来自社で保有すべき中核部分のみに集中したり、事業にあわせて拡張や縮小をしたりして、アジャイルな経営の一助にすることもできる。

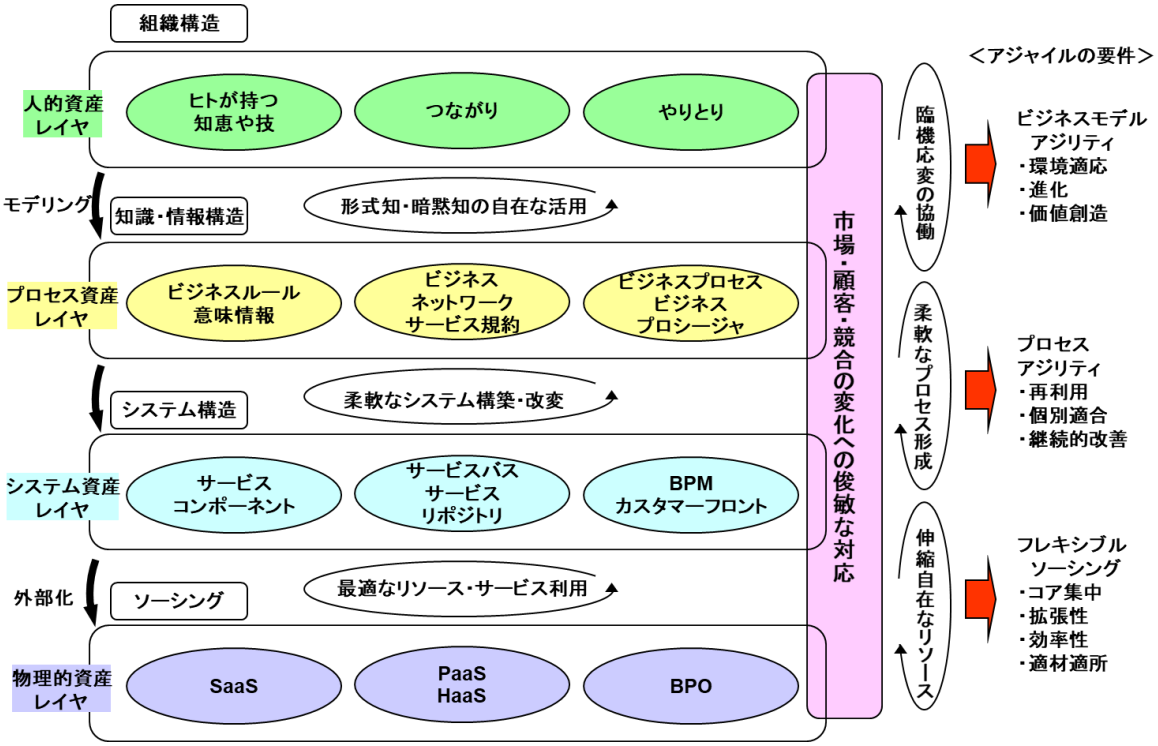


図 7　アジャイルな企業のアーキテクチャ　筆者作成

アジャイルな企業を実現する EA

SOA による業務とシステムのサービス部品化

SOA を実際に運営するには、ビジネスプロセスの構築・改訂のサイクルと、ビジネスプロセスの共通部品であるサービスの構築・改訂サイクルを分ける必要がある。

サービスを提供する側では、サービスごとに、設計（デザイン）、実装（インプリメンテーション）、稼動（エグゼキューション）、評価（モニタリング）、改訂（エンハンス）、そして廃棄といったライフサイクルを回す。
サービスを利用する側では、ビジネスプロセスをモデル化し、利用するサービスを選び、サービスとのサービス利用規約（コントラクト）を決定し、サービスを組み合わせて（オーケストレーション）プロセスを動かし、プロセス改革によるビジネス効果を評価（モニタリング）する。
サービスの全体構成は、ビジネスプロセスモデルと意味的情報モデルからなる SOA 参照アーキテクチャに準拠して整備する。

全社のシステム化案件を統括するガバナンス機能については、改革プログラムのガバナンスとプログラムのポートフォリオのマネジメントを行う PMO（プログラムマネジメントオフィス）の機能に加えて、サービスのライフサイクルのガバナンスとサービスのポートフォリオのマネジメントを行う組織機能が必要になる。これを SMO（サービスマネジメントオフィス）と呼ぶ。

SOA においては、論理的なサービスはインテグレーションサービスを介して物理的なモジュールと連結される。これによって、物理的なモジュールの所在が変更されても、論理的なサービスは変更を意識せずに引き続き利用することができる。
SMO は、サービスをどんな物理的リソースに実装すればよいかについても、管理機能を果たす。
サービスをどんな機器上のどんなシステムに実装するか、そのシステムはどのような形態（既存システム、パッケージ、新規コンポーネントなど）で調達するか、ベンダーはどこにするか、実装した先のサービスレベルをいかに確保するか、といった管理である。これをセレクティブ・ソーシングと呼ぶ。
実装するシステムは、Saas（ソフトウェア・アズ・ア・サービス）のような、自社では所有していないリソースを利用する可能性もある。
さらには、システムではなく、外部のビジネスプロセスをサービスとして利用する BPO（ビジネスプロセス・アウトソーシング）まで、選択肢を拡張して考えることもできよう。

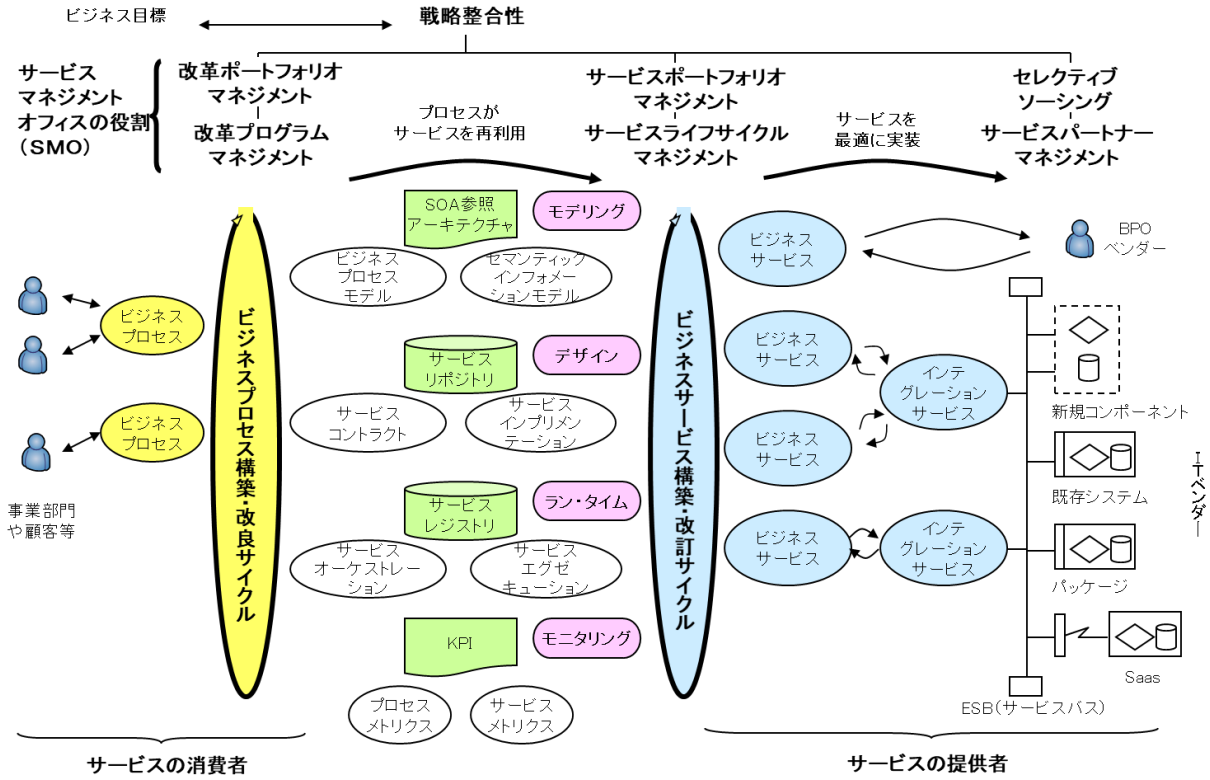


図 8　SOA による業務とシステムのサービス部品化　ローゼンをもとに筆者作成

アジャイルな企業へのメソドロジーの進化

EA を全体の枠組みとしながら、システムの構築においては、ビジネスモデリング、業務プロセス設計、システム設計、システム構築、プロジェクトマネジメント、IT サービスマネジメントといった、仕事の工程に応じて、いくつかの方法論が利用される。

・ビジネスモデリングでは、UML（ユニファイド・モデリング・ランゲージ）というモデル記述手法を使って、業務の姿やデータの構造を、定義された形式にそって図に描くことが一般化している。さらに、UML で詳細な処理の仕方まで表現すると、それを自動的に変換して、実際に動くシステムの部品を生成することができる。こうしたモデルを基にしてシステムを生成する方法を、MDA（モデルドリブンアーキテクチャ）と呼ぶ。

・業務プロセス設計では、業務の流れを描くためにワークフロー図（業務フロー図）が、従来からよく用いられてきた。最近では、業務プロセスをモデル化してわかりやすい図に表現し、図に描かれた業務プロセスをもとに、実際のシステム化された業務の流れを制御しモニタリングする、BPM（ビジネス・プロセス・マネジメント）が導入されるようになってきた。BPM 内に定義されている仕事の流れにそって、サービスとして定義された業務機能を BPM が呼び出して、連結して稼動させることによって、業務プロセスが実行される。

・システム設計では、実世界で行われている業務機能をシステム機能に置き換えていく機能中心設計と、実世界のなかにある情報をデータ構造としてとらえてシステムに映すデータ中心設計が行われている。

この基本的な考え方に変わりはないが、SOA は、業務機能や情報を、実際にビジネスの中で利用できる大きな粒でサービスとしてくくって定義し、サービスを企業全体の業務プロセスで再利用していくことを目的とした設計方法である。

・システムの構築では、業務要件を確定し、システム機能を決定し、プログラムを作り、テストして、実際に稼動させる、という工程を踏むウォーター・フォール型開発が行われてきた。これは、工程ごとにアウトプットを確定させて、次の工程のインプットとし、あらかじめ決めた計画どおりに全体の工程を順々に実施することを前提とした方法論である。こうした工程の定義にもとづいて、システム構築のプロセスを標準化し、継続的に改善して品質を高める方法として、CMMI（ケイパビリティ・マチュリティモデル）が普及している。

こうしたかっちりとした方法論は、今後も有用である。一方で、あらかじめ要件が確定できにくい新たな業務やシステムの開発においては、緩やかな仮説のもとに、まずは目に見える試作品のシステムを作って、それを検証しながら改訂を繰り返していく、というアジャイル開発の方法論が注目されるようになった。プロジェクトの性格によって、この2つの方法論を使い分けることも必要である。

・プロジェクトマネジメントでは、システム構築などのプロジェクトを管理するための標準的な手法として、PMBOK（プロジェクトマネジメントのナレッジ体系）が世界的に普及している。これに加えて、システム構築のプロジェクトだけでなく、それと対になっている業務改革も含めてプログラムとしてとらえる管理、さらには、全社のプログラムの集合をポートフォリオとしてとらえて、全体のやりくりを機動的に行う管理も重要であり、プロジェクトマネジメントも、こうした「3つの P」 のマネジメントに拡張されている。
SOA においては、この「3つの P」 に加えて、さらに共通サービスについてのライフサイクルマネジメントとポートフォリオマネジメントも必要になる。

・構築されたシステムの利活用段階において、最適な IT サービスを提供するためのマネジメントの方法論として、ITIL（IT インフラストラクチャライブラリ）が普及している。これについても、SOA におけるサービスライフサイクルの考え方の反映や、拡大が見込まれるクラウドコンピューティングなどにおける、自分で保有しないシステムを使った IT サービス提供の管理への拡張が、新たに必要になっている。

UML でモデルを描いて、業務フローを書き起こし、機能中心でシステムを設計し、ウォーター・フォール型でシステムを構築し、PMBOK にそってプロジェクトを管理して、出来上がったシステムは ITIL にそって運用する。
これは従来からのシステム運営におけるベストプラクティスである。今後も、こうした方法論に磨きをかけていくことは重要である。特に、安定的なシステム全体構造を持ち、「ワン IT」を指向するような企業では、こうした方法が引き続き有効である。
一方で、業務もシステムも、環境の変化に応じてあるいは自社が持つ多様性に応じて、柔軟に変えていく必要がある企業、すなわち「リユース IT」を指向する企業では、従来からの方法論を拡張して、MDA、BPM、SOA、アジャイル開発、3PM、クラウドコンピューティングといった要素を注入していくことが必要になってくる。

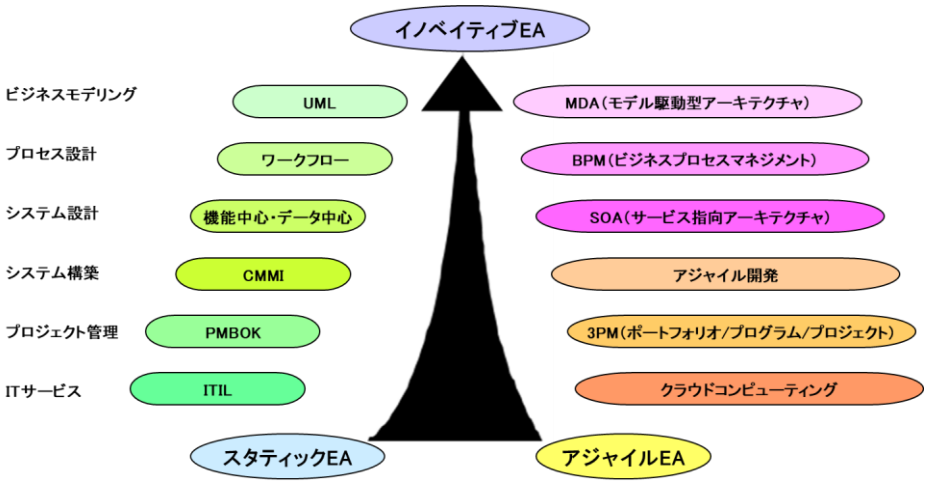


図 9　アジャイルな企業へのメソドロジーの進化　筆者作成

連邦政府での EA 失敗の教訓

アジャイルな企業をつくるためには、その拠り所となる新たなメソッド（方法論）が必要になる。先進的な方法論を採用して、SOA に基づくシステム再構築に取り組んだ事例として、米国の連邦政府機関をとりあげる。この機関は、調達、事務管理、不動産およびその他のサービスを、他のすべての連邦政府機関へ提供している。この機関では、意欲的なプロジェクトの推進や斬新な方法論の導入にもかかわらず、プロジェクト自体は、ユーザーとベンダーの抵抗や上層部のスポンサーシップの喪失によって、挫折の憂き目にあった。この事例はマイケル・ローゼンの調査に基づいている。

基幹業務システム再構築の経緯

1970 年代初め、この機関は請求、資産会計、受取債権の機能を含む、財務システムを構築した。このシステムは本政府機関の基幹業務システム（コアシステム）として、その後も機能拡張を続け、長年に渡って稼働している。コアシステムは、多数の他のシステムやビジネスプロセスと組み合わさって、年間で数十億ドルにのぼる取引を取り扱っている。
コアシステムを置き換える理由は明らかだった。コアシステムの内部や周囲で運用されている業務プロセスとルールが、標準的な会計慣行と合わなくなってきたのである。そこで、市販のソフトウェア・パッケージに SOA インターフェースを組合せて、この計画を進めることになった。

ベンダーに依存しない新システムへの転換

システムの置き換えにおいて重視されたのは、特定のＩＴベンダー1 社へ依存する従来の体質を改めることだった。これまでは、大手ＩＴベンダー1 社が、コアシステムとインターフェースの一切を導入し維持管理していた。新しいシステムでは、他の政府機関や取引先が接続するための、単一の標準的な方法を提供することにした。新システムには次の三原則があった。

・この政府機関システムへの全てのインターフェースは、さまざまな業界のベンダーのだれもが自由に入手でき、かつ導入できる標準に準拠する。

・この政府機関は専用ネットワークを使わず、インターネット上で機密保持されたシステムの運用を行う。

・一切のシステムの仕様は、特定のＩＴベンダーのプラットフォームに依存しないものにする。

この原則に沿って、政府機関は、「モデルベースの取得、サービスベースの調達（Model Based Acquisition、Service Based Procurement）」と呼ぶアウトソーシングの方針をとった。
政府機関のアーキテクトは、導入する必要のあるシステムの論理的な仕様として、ハイレベルなモデルを作成する。このモデルはそれを実現するシステムの RFP（提案要請書）の基礎となる。RFP には、相互運用性を確実にするために必要な具体的なインターフェースを示す。これにもとづいて、パッケージベンダーやシステム統合のベンダーが、システムの導入を行う。

この原則に沿うと、SOA の採用は当然のものであった。
「SOA によって、システムの相互運用性を高めるために、独立したモノリシックな（ひとかたまりの）アプリケーションを 1 対 1 で結合するのではなく、ゆるく結合した（疎結合の）分散サービスを実現する」。

現行のビジネスプロセスとシステムの評価

プロジェクトの第一歩は、コアシステムとそれを取り巻くシステムやインターフェースが、何をしているのかを理解することであった。この機関は、まずあるがまま（as-is）の状況を理解し、その上で、組織の使命、目標、戦略とそれを照らし合わせて、置き換えるべき機能のすり合わせを行うことにした。そして、典型的な EA（エンタプライズ・アーキテクチャ）の方法論に沿って、次の工程を実施した。

① 現行の財務機能の詳細仕様の理解

② 組織を横断するビジネスプロセスのマッピング

③ 現在の IT アーキテクチャの評価

④ 置き換えるべきサービスの概要設計

⑤ 置き換えるべき現在のアプリケーションおよびデータ・アーキテクチャの詳細な評価

⑥ 移行計画とロードマップ（行程表）の策定

移行計画では、新機能を段階的に配備し、コアシステム内の古い機能を一度に一つずつ稼働停止していくことになっていた。プロジェクトでは、この移行方法の実行可能性を評価し、コアシステム全体を最終的に置き換え稼働停止させるための、システム導入手順を確立するはずであった。

現行システムの全体構造を理解するためには、組織構造の理解が前提となる。各部門がそれぞれどのビジネス機能を利用しているか、またそれに対応するシステムはどれかが特定された。これによって、部門、ビジネス機能、システムの、高レベルの機能横断的マップが作成された。

現行の業務とシステムに関する評価には 45 日間しかかからなかったが、明らかになった情報はその後の活動にとって重大な影響を与えた。

・重複したビジネス機能や関連システムが多い。

・重複したシャドーシステム（非公式なシステムのこと）を用いて、部門をまたがって重複したプロセスを実施している。

・コアシステムとのインターフェースが 100 個余りあり、その中には明確に定義されたものがほとんどない。コアシステムを置き換えると同時に、他のシステムも組み込むか、置き換える必要がある。

・新システムへ円滑に移行するためには、重複プロセスのすり合わせや、関連するシャドーシステムの除去が不可欠である。

・ある部門は他部門と協力する意欲がほとんど見られない。コアシステム置き換えプロジェクトのスポンサーと協調することも嫌がっている。

新アーキテクチャの詳細分析

この機関は評価結果をもとに、EA に基づいた方法論を継続し次の工程を実施した。

①テクニカル・アーキテクチャ（IT 基盤構造）の評価
②データ・アーキテクチャおよびインターフェースの評価
③アプリケーション・アーキテクチャ（業務システム構造）の評価
④ビジネスロジック（業務機能）分析
⑤新アーキテクチャ・マッピング

データ・アーキテクチャの評価においては、コアシステム内のプライム（主要な）データ候補のうち、新アーキテクチャ内へ引き継がれる必要があるものの特定に焦点を絞った。そして、システム内でこれらのデータの定義が一貫性を欠いていることが判明した。それに加えて、コアシステムとデータを送受するインターフェースが、100 個もあることが厄介であった。

アーキテクチャ・チームは、この評価段階で収集したすべての情報を、新システムにおける共通サービス候補を特定するための作業に用いた。共通サービスを設計する上での原則は、「ひとつのことをする方法は一つだけ」と「どのプライムデータについても責任を負うのは一か所だけ」であった。

チームはビジネス・アーキテクチャ（業務構造）とデータ・アーキテクチャの策定に着手した。この段階においては、MDA（モデルドリブンアーキテクチャ）の方法論を用いて新サービスを設計した。

まず、この機関の業務機能とそれを実行する機関内部の役割が特定された。次にこれら役割の間のプロセス、情報フロー、アクティビティ、およびサブアクティビティがモデル化された。さらに、これらのモデルから、XML スキーマ、Web サービス、Java コンポーネントといったシステムの部品群が作り出された。

この MDA の方法論によって、ビジネス・アーキテクチャからサービス・コンポーネントへの直接の関連付けが可能となり、コーディングする量が劇的に減少し、論理的に定義されたサービス・インターフェースと実装されるサービス・コンポーネントとの間の一貫性が確保される。

システム移行第一弾の準備

移行の方法は、一度に一機能ずつコアシステムから外していくことであった。最初の移行の対象となった機能は資産会計である。プロジェクトは、新たな資産会計システムを導入し、コアシステム内の対応する機能を停止させるための、一連のシステムを提供した。
提供されたシステムには次のものが含まれる。

・コアシステムの資産会計機能の代わりに配備されたソフトウェア・パッケージ

・コアシステムへの SOA インターフェース

・資産会計データをパッケージのサービスへ、その他のトランザクションを現行のコアシステムへ振分けるインテリジェント・トランザクション・ルーティング機能

一方で、新システムへの移行にむけた組織面の整備はとくには行われなかった。アーキテクチャ・チームは、新アーキテクチャの設計と MDA 方法論の導入については、異常なほどに積極的であった。しかし、「モデルベースの取得、サービスベースの調達」というアーキテクチャ・チームの方針に対しては、利用部門もＩＴベンダーも懐疑的な態度を示した。あるＩＴベンダーはこう言った。「アーキテクトは何もしない。彼は我々に対して、モデルとインターフェース仕様を示すだけで、何を実装するか具体的に指示することができない。」

新システムへの移行の中断

結果、新システムの機能は意図したとおりに動いた。しかし、コアシステムのユーザーからは利用することを拒否された。その理由は、プロジェクト実施中に生じた CIO の交代によるスポンサーシップの変化、進歩的なソリューションについての利用部門の理解不足、外部ＩＴベンダーからの激しい抵抗などであった。

このプロジェクトでは、優れたリーダーシップとリーダーシップの欠如の両面があらわになった。アーキテクトは優れたリーダーシップを発揮し、極めて複雑な現状の課題に対応し、進歩的なシステムソリューションを提供した。今にして思えば、先進的な MDA 方法論に関する利用部門への説明方法については、アーキテクトはもっと改善する点があったかもしれない。
一方で、トップレベルと中間レベルでリーダーシップが欠如していた。プロジェクトの途中で CIO が異動になり、その時に空白が生まれてしまった。くわえて、この穴を埋めるべき中間レベルの管理者層は、既存の外部ＩＴベンダーの圧力に屈し、昔からの仕事のしかたへ戻ってしまった。新たな CIO も、先進的な方法論をもはや支持しなかった。

結局プロジェクトは頓挫した。別の置き換えアプローチが検討されたが、現行のコアシステムは、その後も利用され続けることになった。

プロジェクトは何故失敗したか

米国政府機関は、新たな方法論の導入に挑戦した。EA をはじめてとして、MDA や SOA といった導入した方法論自体は、先進的でありかつ目的にそった妥当なものであった。しかし、プロジェクトは頓挫した。何故か。

だいいちに、プロジェクトが、あくまでシステムの最適化だけにとどまった範囲で行われた

ことだ。システム化の前段として、現行業務プロセスの評価は行われたが、それはあくまで、新システムの機能を設計するためであって、業務プロセスの改革や、ましてや組織機能の改革は行われなかった。システムだけ最適化しても、ビジネスが最適化されなければ、不整合が生じてあたりまえである。EA も SOA も、ビジネスとシステムの両面での全体構造の改革であることを、改めて強調しておきたい。

また、経営層の関与があいまいな中で、理想的な方法論だけが先走りをし、組織全体の中で、プロジェクトが宙に浮いた存在になってしまったことも問題である。システムの利用部門の理解や、参加するベンダーの賛同も得られないなかで、アーキテクトだけが理想論を振りかざしてみてもことは進まない。実施現場の事情を軽視した方法論先行の進め方がアダとなったといえる。

そして、一気に先進的な方法を導入するにしては、組織の実行能力がそこまで成熟してはいなかった。数十年使ってきたシステムに慣れている IT 部門、ベンダー、システム利用部門にとって、いきなり MDA だ、SOA だといわれても、サービス提供者としてもサービス利用者としても意識や能力が追いつかないのだ。
これまで自分の使い勝手が良いシャドーシステムを使っていた利用部門に、これからは、共通システムの「リユース」だ、といっても何のメリットが感じられるだろうか。小さく初めて成功例を作ってから段階的に拡大するという、利用者を巻き込むための「バリューベース、デマンドドリブン」の展開戦略をもっと慎重にとるべきではなかったか。

また、オープンな調達や特定のベンダーに依存しないシステムという理想はすばらしいとしても、実際には、誰がシステム全体の導入に責任を持つのだろうか。ベンダーのいいところ採りをするセレクティブ・ソーシングは、ベンダーを使いこなす側の強い統治力がなければ成立しない。そもそも、調達のオープン化が目的で SOA を採用することが妥当といえるか。SOA の採用目的はあくまで業務とシステムの柔軟性確保であろう。この機関のシステムでは、もっとソリッドなアーキテクチャでも良いのではないか。

つまり、メソッドの問題の前に、プロジェクトのスコープ（範囲設定）、スポンサーシップ、ガバナンス、妥当なアーキテクチャの選択、利用者と提供者のケイパビリティ（実行能力）といった、他の要素が、この機関の場合は不足していたといえよう。
メソッドは改革を進める上で大きな拠り所にはなるが、あくまで手段を提供するだけである。メソッドだけが先行することにならないように、ガバナンス、アーキテクチャ、ケイパビリティも含めた４つの方策のすべてが、整合性を持って実施されてこそ、初めてメソッドも威力を発揮することができる。

EA の成熟段階に応じた 4 つの方策の実施

ITによる変革で何を実現したいのかというIT 活用目的が高度化していくのに対応して、システムの全体構造は成熟度を高めていくものと考えられる。そこであらためて、ジニー・ロスによるシステムの全体構造の成熟段階の考え方に対応させて、EA の成熟段階を、次の5つに定義しよう。

① 個別業務課題の解決（ローカルIT）・・・「個別最適型」に対応
自社内にある各部門の個別の業務課題を解決するために、ITを用いて個々の商品、サービス、プロセスを改善する。

② システム効率性の向上（シェアドIT）・・・「IT基盤標準型」に対応
これまで個別に作ってきた業務システムの、使いにくさや維持管理・運用の非効率さが問題になってきたので、IT基盤を刷新し、標準的な共通化されたIT基盤の上で各業務システムを効率的に稼動させる。

③ 共通業務プロセスの改革（ワンIT）・・・「プロセス・データ統合型」に対応
他社に優る業務の品質、生産性、スピードを実現しオペレーショナル・エクセレンスを獲得するために、ITを用いて部門をまたがる共通の業務プロセスを全社最適化する。

④　ビジネスモデルの改変（リユースIT）・・・「共通部品型」に対応
事業環境の変化に即応できるように、再利用可能なIT部品を用いてビジネスモデル（事業構造や収益を生み出す仕組み）を俊敏に改変できるようにする。

⑤新ビジネスの創造（イノベイティブIT）
グローバルに通用するような自社の独自の価値を生み出すために、ITを用いて新しいビジネスを進化させ続ける。

アジャイルなビジネスを目指す場合には、「ワンIT」や「リユースIT」の段階のIT活用が行われている。単一のビジネスモデルを全社に展開し一気に拡大させることに重点を置く企業では、「ワンIT」が IT 活用面でのゴールとなる。さらに、多様な事業を抱えそれぞれの自律的な発展を目指す企業では、「リユースIT」 の段階まで進む必要が出てくる。5 段階目の「イノベイティブIT」は、再利用可能になったシステムを進化させ続けて、新たなビジネスを創造し続けていく「リユース IT」 の発展型を意味している。

IT の段階は、順を追って進化させる必要がある。ローカル IT の企業が一気にワン IT やリユース IT を目指すことは困難である。

・まずシェアド IT を達成することによって、アプリケーションやデータを共通の IT 基盤に載せて統合化に向けた準備をする。
・ワン IT を実現すれば、業務とシステムの全体構造に関する青写真が現実のものとなり、共通部品化の下地となる。
・リユース IT を実現すれば、共通部品を持続的に改変したり追加したりして進化させ続けるイノベイティブ IT の環境が整う。

システムの全体構造(アーキテクチャ)の進化と同期をとって、管理・統制(ガバナンス)、手法・技術(メソッド)、組織人材(ケイパビリティ)も進化させる必要がある。
・シェアド IT では、業務改善やプロジェクト管理の方法を標準化し、標準に沿って各現場が主導で個々の改善を進める。
・ワン IT では、企業としての戦略整合性を確保するように変革プログラム全体を管理し、組織全体で変革を推進する人材を育成し活躍させる。
・リユース IT では、共通サービス部品の運営を管理するサービスガバナンスを確立し、MDA や SOA といった共通部品化の手法を適用し、各事業部門が共通部品を利用して自律的に事業を成長させる。
・イノベイティブ IT では、新ビジネスを創造するプロセスを確立し、イテレーション(試行錯誤)やプロトタイピングの手法を適用し、創造性を重視した人材育成を行う。

WHEN:成熟段階		WHY:管理・統制	HOW:手法・技術	WHAT:全体構造	WHO:組織人材
EAの成熟段階	何を改革するか	ガバナンス	メソッド	アーキテクチャ	ケイパビリティ
部門最適 (ローカルIT)	個別の商品・ サービス・ プロセス	改革の統制未整備 リスクマネジメント (情報漏えい対策のみ)	業務改善サイクル (継続的な改善) エンジニアリング (CMMIレベル2)	個別最適型	コミュニケーションのみ
IT基盤 全社最適 (シェアドIT)	標準IT基盤	ITプロジェクトマネジメント リスクマネジメント (アクセス管理)	業務改善サイクル エンジニアリング (CMMIレベル3)	IT基盤標準型	技術スキル標準定義 現場主導の能力発揮環境
プロセス 全社最適 (ワンIT)	全社共通プロセス	戦略整合性の確保 改革プログラムマネジメント リスクマネジメント (可用性・正確性)	トランスフォーメーション (抜本的改革) エンジニアリング (CMMIレベル5)	プロセス・データ 統合型	組織的な成長機会の設定や能力発揮環境
グループ 全体最適 (リユースIT)	ビジネスモデル	サービスポートフォリオマネジメント サービスライフサイクルマネジメント リスクマネジメント (不確実性への対処)	MDA・SOA (俊敏なビジネスモデルの改変)	共通部品型	自律的な成長を尊重
ビジネス創造 (イノベイティブIT)	新ビジネス	創造プロセスマネジメント リスクマネジメント (潜在的な機会・脅威)	ビジネスプロトタイピング (ブレークスルー)	進化し続ける 業務システム	創造性を重視

図 10　EA の成熟段階に応じた 4 つの方策の実施　筆者作成