要求の変化に対応する情報システム構築技術の適用に関する調査

調査報告書

2012 年 4 月 26 日

独立行政法人情報処理推進機構

技術本部 ソフトウェア・エンジニアリング・センター

はじめに

IPA（独立行政法人情報処理推進機構、理事長：藤江　一正）技術本部　ソフトウェア・エンジニアリング・センター（以下、SEC）では、日々変化する社会環境やビジネスニーズ等に柔軟に対応するための情報システム構築技術の適用に関する指針づくりに向けた調査に取り組んできました。当報告書では、要求の変化に対応する情報システムの構築に関する技術課題を収集し、ステークホルダー要求の観点などの切り口から情報システム構築技術の体系化を行い、その調査結果をまとめました。

本調査事業は、「要求の変化に対する情報システム構築技術の適用に関する調査事業」として、株式会社三菱総合研究所に委託し、実施しました。

要求の変化に対する情報システム構築技術の適用に関する調査
【調査報告書】

独立行政法人情報処理推進機構

目　次

1. 調査の背景と目的

1.1 調査の目的

本調査では、「要求の変化」という課題にとって重要となる技術項目を調査し、技術課題をまとめる。変化が激しい現代社会において、社会や企業を支える情報システムを構築するための重要な技術を整理し、提供することで、我が国の社会インフラや企業活動の強化に資すること、特にユーザ企業における情報システム構築の技術力、産業力強化に資する技術的な指針が示され、刻々と変化する社会や国民ニーズに対応可能な IT 社会の実現に資することを目的とする。

本報告書全般に関する注意

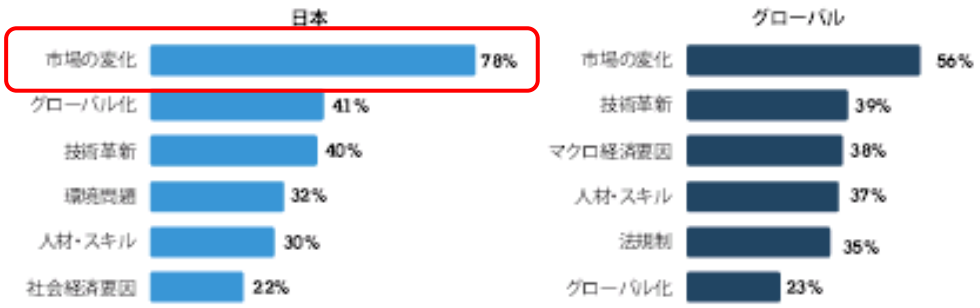
本報告書で活用している多くの用語について、日本語訳が定まっていないか、JIS として定義が進められているものなどがあり、日本語として訳することが不適切なものも多い。読み易さのために日本語訳を試みているが、ここで使われている日本語は本報告書限りとする。各種団体による標準化が待たれる。

1.2 調査の背景

(1) 様々な環境の変化

企業や社会にとって IT 活用の重要性は年々高まっている。特に情報システムは、企業の業務システムや制御システムのみならず、CPS (Cyber Physical Systems) の発展で組込みシステムにとっても活用される場面が増えてきており、逆にスマートフォン等の組込みシステムから気楽に情報システムがアクセス可能となり、国民からも身近な存在になっている。

他方、2010 年に日本 IBM が配付した「IBM Global CEO Study 2010 Japan Report」（[10]）にあるように経営者が考える自社に大きな影響を与える外部要因として「市場の変化」を挙げている（図 1）。



出典：日本 IBM： Global CEO Study 2010 Japan Report

図 1　今後の 3 年間で自社に大きな影響を与える外部要因

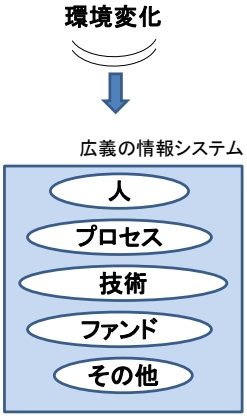
実際、社会システムを取り巻く環境が、安定成熟化・少子高齢化・価値観の多様化等、目標達成型から個人嗜好追求型社会へと予想を超えた速度で変化している。一方でクラウド化やスマートフォンの普及等、ネットワークを介した複数のシステムが絡みあったサービスの登場等「技術の変化」もみられる。「市場の変化」が我が国の企業を襲っているのが現状である。我が国が、政策的に IT 化を進めることで企業の競争力を高くしようとしている状況下では、情報システムの構築は「市場の変化」、「技術の変化」に追従することを求められていなければならなかったと考える。

さらにギリシア危機の影響による円高の高止まり、さらに 2011 年 3 月 11 日の震災やそれを発端とした電力不足、タイの洪水などの自然環境の変化の影響など供給側も含めた市場環境、経済環境も大きく変化している。高齢者や失業者の増加、震災対策等も含めてネガティブな意味でも社会環境は絶えず変化をしている。環境の変化に追従した事業経営や社会サービスが求められる状況下で、これらの環境の変化に追従して支援し続ける情報システムが求められる時代と考えることができる。

環境の変化を被るのは、モノとしての狭義の情報システムではなく、人、プロセス、技術さらにはファンドやその他諸々のシステム要素で絡み合って構成されている情報システム（本報告書では「広義の情報システム」と呼ぶ[1]）であることに注意されたい（図 2）。環境の変化はステークホルダー要求の変化に帰着される。

---

[1] 要求の変化をとらえるとき情報システムだけでなく、それを利用・活用・運用する全体活動を対象とするため、暫定的にここでは「広義の情報システム」という用語を使う。正しい用語は Enterprise であるが、誤解を招く恐れがあるため本用語は使わない。



文献[1]の Fig2-2 を参考に作成

図 2　環境変化の影響を受ける「広義の情報システム」

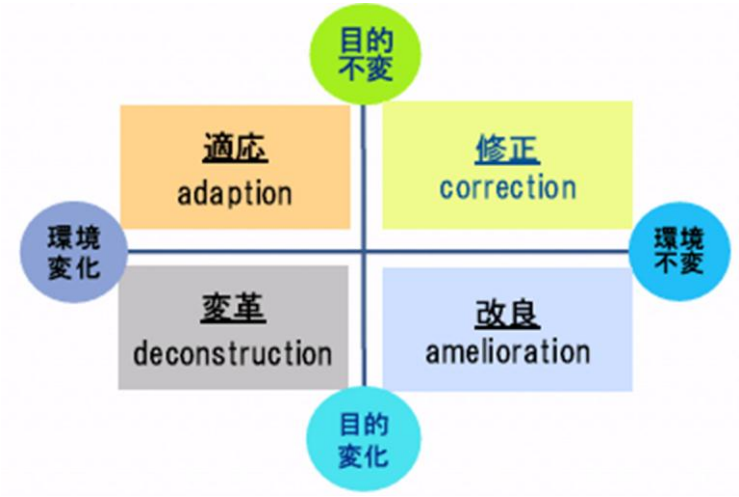
環境の変化が激しい時代にも関わらず、既存の多くの情報システム開発では、V 字型開発で描かれるプロセスフローの意識が高く、環境の変化にシステムを対応させる必要性が生じた場合、その対応は開発を完了したシステムの保守・運用プロセスによるマネジメントの一環として実施される。

環境の変化は、演繹的にはステークホルダー要求の変化としてとらえることで、システム保守のみならず IT サービス（運用）を含むビジネスプロセスの変化も伴う必要がある。システム開発工程だけではなく、超上流工程での検討も要することになる。

環境の変化の激しい状況への対応を、適用保守として実施し続けることには無理があると考える。「変化し続けること」と対象となっているシステムのシステム化計画への影響性を明確にすべきであり、考え方を再整理しておく必要がある。

コラム　要求変化の類型化

要求の変化の種別分けの研究がある。（独）情報処理推進機構（以下、IPA と略す）技術本部ソフトウェア・エンジニアリング・センター（以下、SEC と略す）エンタプライズ系プロジェクト要求発展型開発ワーキンググループ（以下 WG と略す）主査の名古屋大学山本修一郎教授は要求の変化をのとおり類型化している。今回の適用範囲は、図 3 の第 2（左上）、第 3 象限（左下）である。



出典：山本：高信頼性システムのアプローチ・チェックポイント、SEC 特別セミナー「アーキテクチャ指向エンジニアリングと形式手法」資料, 2011.

図 3　要求変化の類型

(2) 持続可能性と要求の変化

「持続可能性（サステナビリティ）」という言葉がある。数年前までは「持続可能性」は自然環境との共生が主なテーマであったが、現在では次のような場合にも使われる。

- 2011 年 3 月の震災やその後のエネルギー不足、タイの洪水、ギリシア危機と円高の高止まり等々、想定外の先の見えない状況下である。
- このような状況下で市場環境、自然環境、さらに社会環境の変化に対応できる社会システムや事業経営を持続的に運用していく。

そのような社会や企業の事業経営を支援するためにも、これらの変化に追従できる情報システムを提供可能にする必要がある[2]。様々な環境の変化に対応するために要求の変化に対応していく情報システムの構築方法は、一般的なシステム構築とは開発方針が異なることになる可能性がある。例えばアジリティの高いシステムの構築はすでに、重要視する開発技術に明白な差異があると考えられる。環境の変化、すなわち要求の変化が激しい時代の情報システム構築技術の在り方を世に示していく必要がある。

上述の IBM のレポートの結果は、環境の激しい変化は事業の経営に影響すると解釈することができる。社会や企業の事業経営者が、激しい環境変化に対応できる持続可能型社会、持続可能型経営を実現するために持続可能な情報システムを活用できることは、ひいては我が国の社会や産業の力を強化することにつながると考える。

2010 年度 IPA/SEC が実施した「保守性の高い情報システムの構築技術に関する調査」（以下、「2010 年度調査」と略す）の調査結果には、技術課題には直接反映されていないが、有識者意見があり、そこでは「現在の急務は現存するシステムを環境の絶えざる変化に合せて進化させながら運用していくにはどうしたら良いかという『システムの維持可能性』（サステナビリティ）を高めること、すなわち Sustainable Software Process for System Evolution を確立することであろう。」と述べられており、さらに「現実世界に存在するアプリケーションは無限のパラメータを含んでいる。要求仕様は、それらのパラメータのうちいくつかを、ある仮説に基づいて切り捨て、有限の形にまとめたものである。時間の経過とともに周囲の環境条件が変化し、最初は無視して良いと考えたパラメータがシステムにとって重要な意味を持ってくることがある。」とも述べている。要求を逸脱した場合に対応可能な運用も含めた体系が求められている。

コラム　　持続可能性とソフトウェア工学

ソフトウェア工学分野において、最も権威ある大会のひとつである「ICSE （International Conference on Software Engineering)2012」におけるテーマが「Sustainable Software for a Sustainable World」である。

持続可能性社会を支えるソフトウェアの実現は、世界的にも重要なキーワードのひとつであることがわかる。そのためにも「要求の変化」に対応する情報システム構築技術をまとめることはソフトウェア工学の世界の成果のひとつとなる。

[2] 2011 年東北地方太平洋沖地震の後、レジリエント（回復性）という言葉が注目を浴びているが、持続可能性と極めて類似した用語である。

(3) 障害発生と要求の変化

技術の進歩によりシステム同士がつながるケースが増えており、システムが複雑化している。そのためシス障害が発生したとしても、その原因が思わぬところに起因していることがあるような状況である。システム障害の発生に応えるためにアシュアランスを確保するための技術の適用が求められる。

要求の視点で障害発生を考察してみる。「障害が発生した」ことは、単純には「システムが要求通りに動かなくなったこと、要求を逸脱した状態になったこと」ととらえることができる。要求逸脱は、動的ではあるが、要求が変化してしまった状況の発生に相当する。障害発生に強いシステムを構築するということは、要求が逸脱しにくいシステムを構築することである。要求の逸脱に対して、例えばインフラを多重化するなど、アーキテクチャによって逸脱を防ぐ仕組みを導入する場合もあるが、多重系の障害への対応も含めて人間系のオペレーションにより回復させることが一般的と考える。つまり狭義の情報システムではなく、人も含めた広義の情報システムとして要求逸脱状態になったことをモニタリングし、その結果を受けて要求を満足する状態へ早期に復帰させる、つまり要求の変化に強い情報システムの構築とは、構築だけではなく利用・運用も含めた技術である。

コラム　　要求の変化と障害対応

（独）科学技術振興機構における「実用化をめざした組込みシステム用ディペンダブル・オペレーティングシステム」研究領域（DEOS プロジェクトと呼ばれる）では、変化に対応するサイクルとして DEOS プロセスをまとめている（文献[13]）。

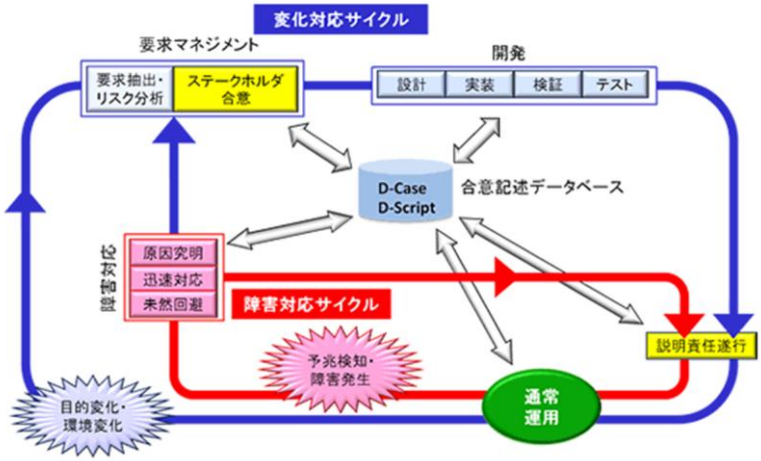


図 4　DEOS プロセス

1.3 情報システムの継続性の実現技術の課題解決に向けて

要求の変化という観点から情報システム開発に注目すると、適用するシステムはすでに継続的に稼働し続けていること、要求管理者が全体を見続けていること、つまりビジネスにおける IT 利用・運用起点（**ビジネス-IT オペレーション**[3]起点）でシステムを想定する必要性がある。

システムを構築し、保守工程で修正を図る、つまり形式的には要求が変化しないことを前提としているが、変化してしまった場合に対処を行う発想、つまりボトムアップ的なアプローチがある。他方、本調査が対象とする絶えず要求が変化することを前提とした情報システムの実現に対しては、図 5 のとおりトップダウン的に要求の変化に対応する情報システムの構築をコントロールするアプローチを仮説として考える。

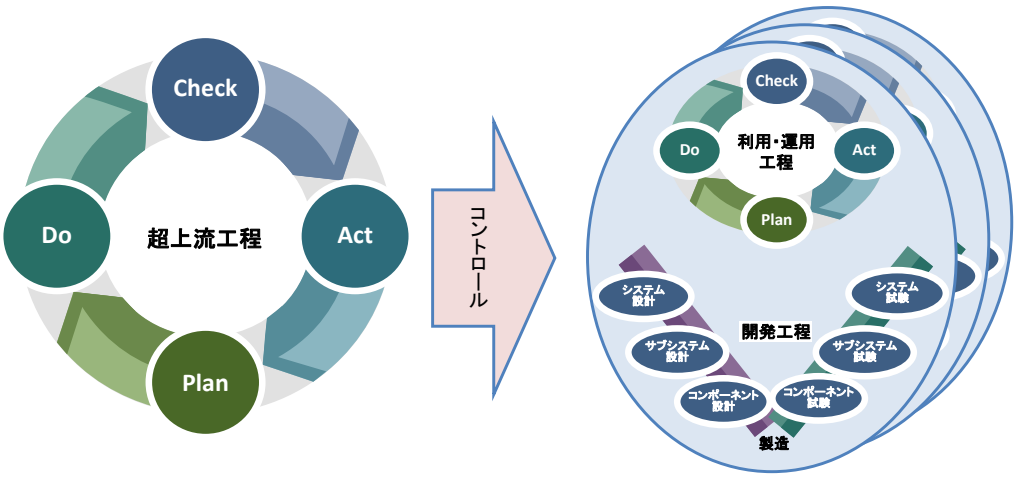


図 5　要求の変化を超上流工程からコントロール[4]

想定外事象の発生も含む要求変化を前提として情報システムの継続性を保つことは、要求の変化を計画的にコントロール[5]することと考えられる。一方で出来上がってから

---

[3] ISO/IEC12207:2008([15])の Operation および ISO/IEC15288:2008([16])の Utilization を併せて利用・運用と記載されている文書もあるが、本報告書では超上流工程の一部も含まれるため、本報告書限定で「ビジネス-IT オペレーション」と記載する。この範疇に用語がないことは今回の課題のひとつである。

[4] 右の開発工程における V 字要素をシステム設計、サブシステム設計としているが、システムの要件定義はシステム設計、ソフトウェアの要件定義はサブシステム設計（階層的である）内に含めている。

[5] 「ガバナンス」という用語に極めて近いが IT ガバナンスとは意味合いが異なるため、区別するためにも本報告書ではコントロールの方が適切と考える。

保守工程だけによる対応を「マネジメント」による対応と呼ぶ。要求の変化への対応は、業務と情報システム群（環境の変化を被るのは単独の情報システムではなく、関係する情報システム全体である）への要求の変化をコントロールする者、要求を管理する者などシステム化計画者を含む超上流工程で登場するステークホルダーが求められる。そしてそれらのステークホルダーが関係するビジネスアーキテクチャを想定した世界をより一層明確に描くことになる。ここで留意することとして、システムの安全性も、システムの「つくり」だけで解決することに限定することは現実的でなく、ビジネス-IT オペレーションとしての対応が求められることが挙げられる。

「要求の変化」、「ビジネス-IT オペレーション起点」としてシステムの実現をとらえると、既存の V 字型のプロセスフローとは違った、新しい情報システムの実現方式としてとらえることができる。活用技術が異なるということではなく、重要と考える開発技術が既存の開発とは異なっているものがあると考える。実際クラウドのようなサービス調達のインテグレーションを想定すると、変化はビジネスおよびシステムの利用運用に従って調達するサービスとサービス運用の変更に帰着できる。

2. 調査手順と調査方法

2.1. 調査項目と調査手順

本調査は、IPA/SEC 内に設置された委員会である「要求発展型開発ワーキンググループ（主査：　山本修一郎　国立大学法人名古屋大学 教授）」（以降 WG と略す）により、調査内容のレビュー結果を受けつつ進められた。そのため、調査内容も当初計画から変更が加えられた。当初計画による各調査項目の調査内容は次のとおりであった。

① 新しい技術課題の追加調査

次の項目における当該開発技術の技術課題（概要、課題解決効果、課題解決に向けた留意点・課題、情報リソース）を調査する。

- システムズエンジニアリング（超上流工程分析評価技術を中心に）
- 高信頼システム（検証技術）
- 非ウォーターフォール型開発技術

さらに 2010 年度調査結果についてフィージビリティを調査し、その結果を反映して、再整理する。

② 技術課題項目の整理と深掘調査

以下の観点に沿った 2010 年度も含めた技術課題の整理を行う。

- ライフサイクルプロセス、ステークホルダー要求
- 品質要求と制約（非機能要求）、アーキテクチャと構成要素

本調査は要求発展型 WG によってレビューを受けながら実施した。WG によるレビュー内容は次のとおりであった。

① 要求発展型 WG 主査との事前打合せ（2011 年 9 月 30 日（金））

- ConOps (Concept of Operation) が重要な位置づけにある。
- 利用・運用を起点としたプロセスフローを想定する。
- 整理軸となる観点をまとめることで、整理軸の相互関係を明確化する必要がある。

② IPA 殿との検討および WG によるレビュー

(a) 第 1 回 WG（2011 年 11 月 8 日（金））

- 基本的な方針を確認

(b) 第 2 回 WG（2011 年 12 月 15 日（木））

- IT サービスを開発との軸の違いを確認、outer loop と inner loop について確認

(c) 第 3 回 WG（2012 年 1 月 12 日（木））

- 整理軸の確認

(d) 第 4 回 WG（2012 年 2 月 14 日（火））

- 技術課題と要求の変化との関係性の整理等
- 報告書内容の確認

さらに IPA/SEC 内での報告会も実施し、レビューを受けた。

- IPA/SEC 報告会（2012 年 2 月 13 日（月））

WG での指摘事項により、各調査項目に対応する調査内容は次のように調整された。本調査報告は、この調整結果に沿ってまとめた。

① 新しい技術課題の追加調査

- システムズエンジニアリング（超上流工程分析評価技術を中心に）
- 高信頼システム（検証技術）
- 非ウォーターフォール型開発技術

2010 年度調査結果の見直しに加え、ITIL 関係技術を運用技術として加える。

② 技術課題項目の整理と深掘調査

- 次の集約した観点による技術課題整理
  - ライフサイクルプロセス

    見える化の観点としてライフサイクルプロセスを適用した。
    - 超上流工程（ESE）
    - システム構築（ISO/IEC12207[15]（JISX0160[21]）、ISO/IEC15288[16]）
    - システム利用・運用（ITIL(ISO/IEC20000[17])）
    - アーキテクチャ開発（TOGAF-ADM）
  - ステークホルダー要求

    今回の技術の分類としてではなく、具体的なシステムが設定できる場合に意義のある方法となるため、整理の例として適用した。
    - ひとつの例として分類を提示
    - さらに分類例として品質特性（ISO/IEC25010[18]および制約）による特徴付を追加

- 山本WG主査の分類は要求の属性として表現
- レイヤは技術の属性として表現
- アーキテクチャは経産省の EA の参照モデルを属性として表現

－ その他

- 品質要求と制約（非機能要求）
- アーキテクチャと構成要素

なお、WG のメンバは次のとおり。

要求発展型開発ワーキンググループ：（以下、敬称略、順不同）

主査：　山本　修一郎　国立大学法人名古屋大学
委員：　赤埴　淳一　日本電信電話株式会社
委員：　秋山　浩一 富士ゼロックス株式会社
委員：　白坂　成功　学校法人慶應義塾大学　大学院
委員：　鈴木　三紀夫　NPO 法人ソフトウェアテスト技術振興協会
委員：　成瀬　泰生　富士通株式会社
委員：　山本　久好　日本アイ・ビー・エム株式会社
委員：　鷲崎　弘宣　学校法人早稲田大学
IPA/SEC　山下　博之
IPA/SEC　吉元　一郎
IPA/SEC　岡山　歩

2.2. 調査方法

(1) 新しい技術課題の追加調査

2010 年度調査に引き続き、技術課題の追加調査を実施した。具体的には図 6 に示す範囲の調査を実施した。

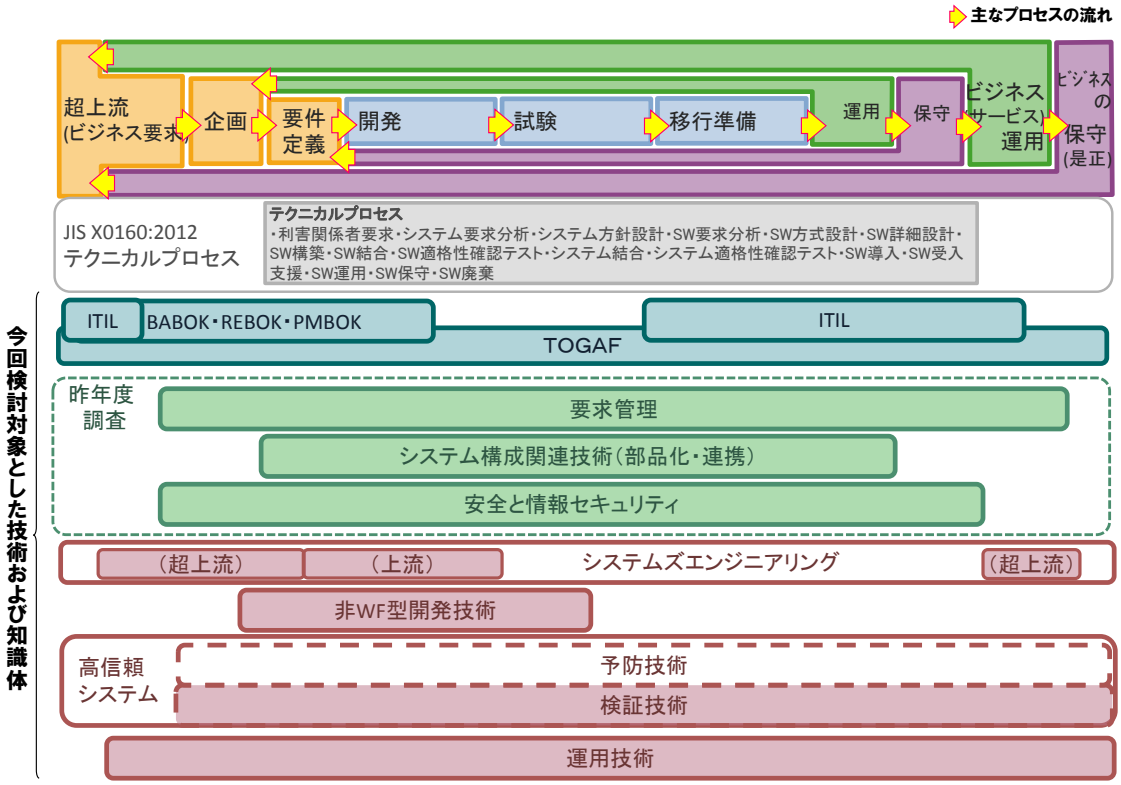


図 6　2011 年度追加調査対象の位置づけ

2010 年度の調査対象が、「保守性の高い情報システムの構築技術」であった。保守性を高くする要因として、情報システムにアジリティの高さや移行性の高さを必要としている状況があった。それらは因果関係で捉えると、結局何かしら「要求が変化した」ことで情報システムの変更の必要性が発生しているということがわかった。そこで 2011 年度は「要求の変化に対応する情報システムの構築技術」、つまり「情報システムの継続性に関する技術」を対象として、要求の変化に着目した技術課題の調査を実施している。

要求の変化という観点に着目すると、因果関係を追うことで要求の根源としてのステークホルダー要求の変化の情報システムの継続性への影響として全体問題をとらえることができる。

そこで、2011 年度調査では、ステークホルダー要求が関係するもっとも上流よりである超上流工程も含めた要求の変化に関わる技術も調査対象とした。また 2010 年度調査で触れられなかった開発技術についても 2 テーマの技術領域について技術課題を求めた。一方で情報システムの継続性を支える要素として利用・運用技術にも技術課題を求めることとした。

(1-1) 新規調査

追加調査として 3 つの分野について調査を実施した。各項目における選定理由と調査方法は次のとおりである。

① システムズエンジニアリング（超上流工程分析評価技術を中心に）

【選定理由】

「要求の変化」に対応する超上流工程を含む技術として、国際的見地でシステム構築の代表的技術であるシステムズエンジニアリングを対象領域として選定した[6]。

システムズエンジニアリングは人的アーキテクチャも含む「システム」として全体を構築する、非常に広い分野を対象としている。そのため超上流工程、上流工程の技術が中心であり、具体的な実装部分はコンポーネントエンジニアリングと呼ばれ、各種開発手法に依存しているため、2010 年度調査とは重ならない領域を対象として含む。

システムズエンジニアリングでは、要求の変化に対応する技術として「エンタプライズシステムズエンジニアリング」がすでに策定されている（[1]）。当該技術を活用する主なプロセスはシステム構築プロセスとビジネスマネジメントプロセスの間に位置付けられる。

システムズエンジニアリングの技術として短期開発や変更容易な開発構築に適したツール化向けプロセス横断的方法論が、システムズエンジニアリングとして最も著名な書物である文献[3]に掲載されている。それらも対象技術として選定した。

背景でも触れている要求逸脱に関係するシステム障害のシステムズエンジニアリングでの対処方法について、文献[2]においてシステムシンキングを活用してまとめられている。

【調査方法】

次のとおりである。

- 文献および Web 調査により技術を洗い出した。
- その結果を専門家により内容の確認を受けた。専門家としては当分野の我が国の代表的研究者である慶應義塾大学大学院の白坂成功准教授（WG 委員）にレビューを受けた。

---

[6] プロセスの一部はシステムライフサイクル ISO/IEC15288 として知られている。

② 高信頼システム（検証技術）

【選定理由】

要求の変化に対応する情報システムは早期にシステム提供をすることが前提となるため、短期間での検証を効率的に進めることが期待されるため、検証技術を選定した。

要求の変化に対応する情報システムは継続性が重要であるため、運用も含めた信頼性を求められる。文献[8]でまとめられているとおり、高い信頼性をもつために予防活動と検知活動が求められている。2010 年度調査において予防活動のための代表的技術として形式手法のみが取り上げられていたが、一方で検知活動として取り上げられたものがなかった。そこで、典型的な検知活動である検証技術について取り上げた。

【調査方法】

有識者ヒアリングにより調査を実施した。

- 情報システム系開発を中心に NPO 法人ソフトウェアテスト技術振興協会の鈴木三紀夫氏（WG 委員）へのヒアリングを実施した。
- 組込み機器を活用したシステムについて富士ゼロックス株式会社の秋山浩一氏（WG 委員）へのヒアリングを実施した。

③ 非ウォーターフォール型開発技術

【選定理由】

要求の変化に対応する情報システム実現のためには、実現における俊敏性（アジリティの高さ）が求められる。そこで当該領域の代表的な技術である非ウォーターフォール型開発技術を選定した。2010 年度調査において注目されなかったため、2011 年度調査により対応した。

【調査方法】

2009 年度、IPA/SEC が実施した「非ウォーターフォール型開発に関する調査」と内容が重複するため、本調査報告書や活用された資料を調査対象とした。

④ 運用技術（ITIL、Information Technology Infrastructure Library）

【選定理由】

要求の変化に対応する情報システムは継続性が求められるため、開発だけではなく利用・運用時の対応も重要視する必要があると WG での議論があり、運用技術のベストプラクティスとして最も有名な技術のひとつである ITIL を調査することとなった。特に ITIL V3 からはビジネス面とのつながりが強調されており、超上流工程の技術としても注目した。

【調査方法】

特定非営利活動法人 IT サービスマネジメントフォーラムジャパン理事である富士通株式会社成瀬泰生氏(WG 委員)より WG の講演にて技術内容を調査した。

WG ではその他の技術に関する討議を実施していたが、特にアーキテクチャフレームワークについても上流工程の技術の一部として、Zachman AF、DoDAF、TOGAF、経済産業書 EA について簡単な調査を行った。しかしながら、技術課題としては取り上げるには至らなかった。一方で EA および TOGAF については技術項目の整理に活用した。

(1-2) 2010 年度調査の再整理

2010 年度調査結果の再整理を行った。当該作業を具体的に次の方法により実施した。

- 2010 年度調査における技術課題と技術項目について精査を行った。
- 分類や名称が、そもそも不適切なものについて技術課題を移動するなどの修正を実施した。説明不足事項については WG 議論を参考に内容の補強を実施した。
- 2011 年度調査内容との関係で必要な項目を追加した。
- WG での議論により 2010 年度調査で取り上げられた技術課題に含まれる技術項目については、当該技術課題に追加した。

(2) 技術課題項目の整理と深掘

整理軸について、WG において検討を行い、再整理を実施した。当初整理軸の設定した上で、再整理結果を示す。再整理は整理の方法が異なるだけで、基本的な内容は変わらない。

(2-1) 整理軸の設定

① ステークホルダー要求別整理

システム要求を変更する場合、実際の変更の必要性および変更の可否判断のためには要求の因果関係をたどる必要があり、そのためにはステークホルダー分析が有効である。技術課題の性質を整理するために、各ステークホルダーの典型的な要求の変化に対する技術課題の貢献の可否を示すことができる可能性があるためである。

② ライフサイクルプロセスに対する整理

具体的に何を実施するところで、技術課題が適合するのかについて整理することで、具体的な開発等での活用の特徴を表現できる可能性があるためである。

③ 品質要求と制約[7]に対する整理

IPA/SEC の非機能要求とアーキテクチャ分析 WG にて非機能要求とアーキテクチャの関係性を調べるために活用した分類方法であるため、技術項目の特徴を表現できる可能性があるためである。

④ アーキテクチャと構成要素に対する整理

同様に IPA/SEC の非機能要求とアーキテクチャ分析 WG にて非機能要求とアーキテクチャの関係性を調べるために活用した分類方法であるため、技術項目の特徴を表現できる可能性があるためである。

(2-2) 整理軸の再整理

前項で洗い出された技術課題の傾向を示すために整理をする観点を検討した。当項目は特に案を策定した後、WG の意見等を取り込んで整理の観点を設定した。調査方法は次のとおりである。

① ライフサイクルプロセスを観点とした整理

- 要求の変化に対応するシステム開発技術に関係するライフサイクルプロセスとして、前項で洗い出された技術項目に対応するプロセスに加え、システム開発の中心である ISO/IEC12207:2008 ([15])および

[7] 制約は主品質特性が決められない非機能要求として定義される。開発標準や例えば Linux 活用など具体的な取り決めなどが挙げられる。

ISO/IEC15288:2008 ([16]) のプロセスのうち必要なライフサイクルプロセスを洗い出す。

- WG にて意見聴取し、その結果を反映
- 次のプロセスを整理軸とし、技術項目をプロットする。
  - 超上流関係プロセス
  - システム構築プロセス
  - 運用プロセス（ITIL）
  - アーキテクチャ開発手法プロセス（TOGAF-ADM）

② ステークホルダー要求を観点とした整理

- 例示に留まるが、具体的に想定するシステムがある場合には特徴付ができるため、整理のフレームとしてまとめる。
- ステークホルダーについて標準類を中心に整理し、本報告書のステークホルダー分類を作成する。
- 当初設定した整理軸について分類が少数なものについては、属性として一項目として表現するべく、整理をする。

なおアーキテクチャとの関係については、結果的には、直接の要素は具体的なシステム開発内容を必要とするため、ここでは経済産業省の EA の属性で留め、TOGAF については、変化を前提とした反復プロセスであるため、①のライフサイクルプロセスへ移動、典型的なアーキテクチャフレームワークとして DoDAF を加え、その観点についての調査結果を「(1) 新しい技術課題の追加調査」の検討結果に反映した。また構成要素については、同様に属性として用意した。

③ その他を観点とした整理

その他として、品質要求と制約に関する整理、アーキテクチャと構成要素に関する整理を実施する。

3. 新しい技術課題の追加調査

2010 年度の調査の視点は保守開発を念頭に置いていたため単体システムのマネジメントに関わる内容を調査した。第 1 章の図 5 を参考に 2011 年度調査をプロットした結果を図 7 に示す。2011 年度は非ウォーターフォールおよび検証技術を中心とした単体システム向け技術に加え、超上流に相当するコントロール系の技術も含む、システムズエンジニアリングや運用技術を取り上げている。コントロール系の技術は単体システムではなく、事業に関係するシステム全体を対象としていることに留意されたい。また、ここで重要なのは、コントロール系の技術か、マネジメント系技術かの選択ではなく、両者の融合が重要になっていることにも留意されたい。

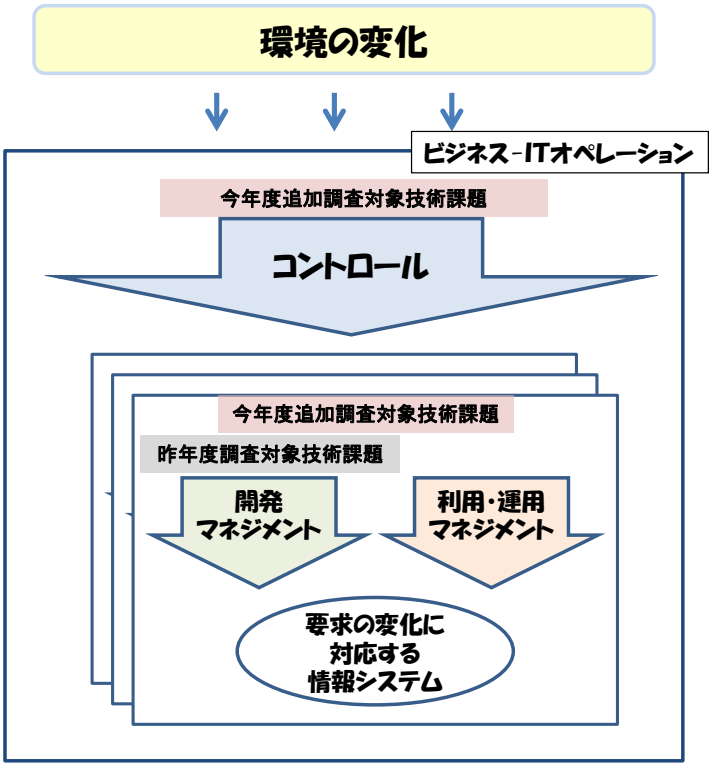


図 7　2011 年度調査の主な範囲と 2010 年度調査の主な範囲（再掲）

3.1. 追加技術課題の調査結果

追加した技術課題について次の項目に沿って説明する。

- 技術課題概要
- 課題解決効果
- 課題解決に向けた留意点・課題
- 情報リソース
- 2010 年度調査視点によるフィージビリティ

2010 年度調査視点とは、「要求の早い変化に迅速かつ柔軟に追従可能な情報システム構築」（以下、「要求の早い変化に追従する構築」と略す）もしくは「移行性の高い情報システム構築」（以下、「移行性の高い構築」と略す）のことである。

3.1.1. システムズエンジニアリング（超上流工程分析評価技術を中心に）

システムズエンジニアリングは、米国の宇宙・防衛関係の技術を中心にシステム構築に向けた技術を体系的に整備したものである。システムとは機器的なものに限定されず、人間世界や、人間と機器、組織、街、また一方でプロジェクトのようなものまで、広く解釈される。そのため、組織的なモノ作りの技術として世界で活用されている。特にシステムとその構成要素であるコンポーネントを分け、システムズエンジニアリングではコンポーネントエンジニアリングについては、活用することに留められており、いわゆる上流工程や超上流工程の技術としてとらえられる。

本項目については、情報システム構築を中心に置いた上で、まず文献・Web 調査を実施して、項目を洗い出し、その後有識者ヒアリングにより内容を確認頂き、次の技術課題についてとりまとめた。

- 超上流工程分析評価技術
- 上流工程における横断的連携技術

前者は、経営とのつながりがあるため、主に分析技術の集合体である。後者は、システムズエンジニアリングの推進組織である INCOSE (International Council on Systems Engineering)が中心になってとりまとめたものが ISO/IEC15288 であり、2008 年度版ではソフトウェアライフサイクルの標準である ISO/IEC12207 と整合性が

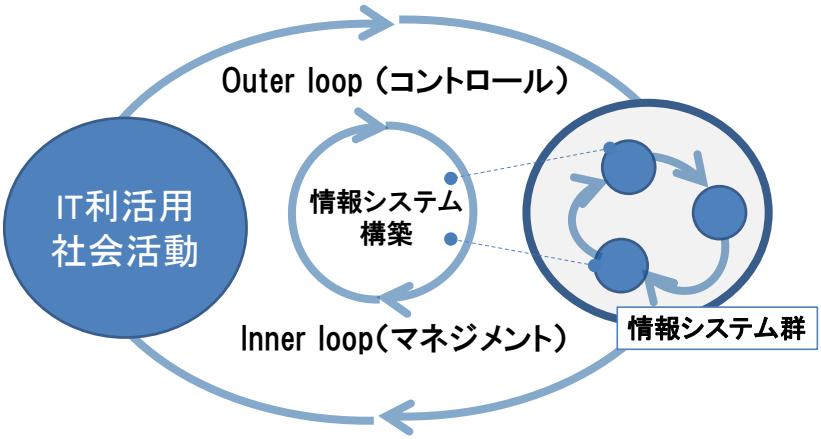
とられており、文献[3]で挙げられた ISO/IEC15288:2008 のプロセスを横断的に連携した方法論をとりあげる。

(1) 超上流工程分析評価技術

(1-1) 概要

前項までで説明したシステム開発のための「コントロール」に関する技術である。

要求の変化を前提とした情報システム構築は、トップダウン指向で構築する技術を必要としている。特に各システム構築（System of Systems を含む）における対応を **Inner loop** と呼ぶことがあり、その場合要求の変化の原因である環境の変化を受け止めるべく本報告書で活用している用語ビジネス-IT オペレーションである超上流工程分析評価技術の改善サイクルは、**Outer loop** と呼ばれる。Outer loop を構成する技術は、必要でかつ重要であるが困難な技術課題である(図 8)。



原出典：Hitchins, Derek, K.: Systems Engineering – A 21st Century Systems Methodology, John Wiley&Sons, 2007.
山本修一郎教授（WG 主査）ホームページ資料を基に作成

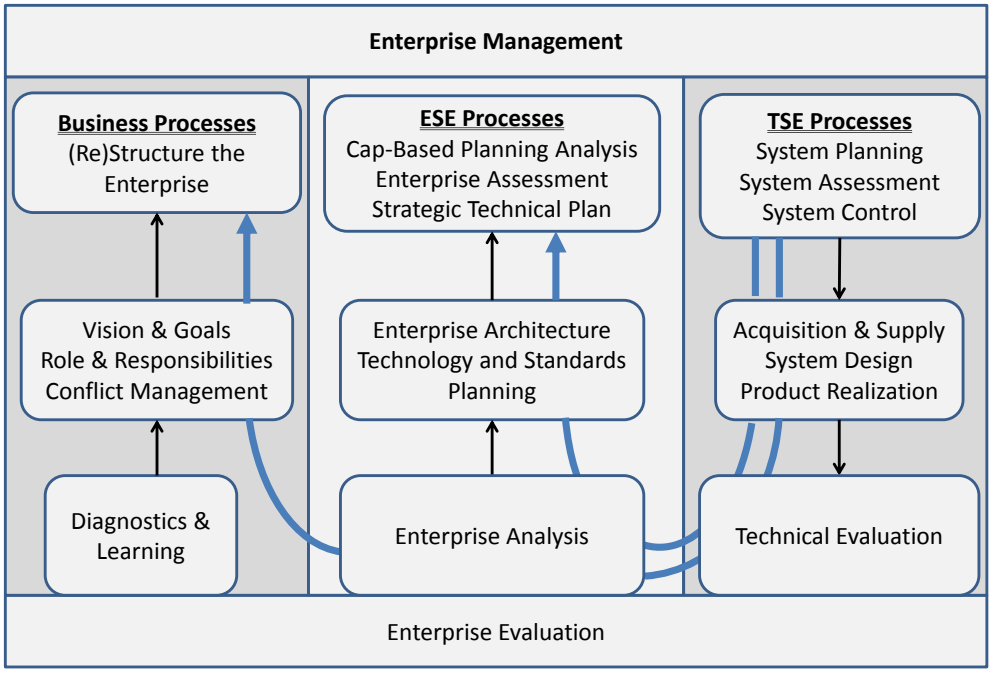
図 8　Outer Loop と Inner loop（システムズエンジニアリングの例）

図 8 では、問題スペースとしての IT 利活用社会活動が環境の変化に伴って変化し、求めたものをひとつの情報システムではなく、利用・運用も含めた情報システム群としての影響と考えて対応し、対応した情報システム群を IT 利活用社会活動が利用していくという意味である。

システムズエンジニアリングにおいて、エンタプライズシステムズエンジニアリング(ESE) が当該内容を担っている。ESE では、要求の変化の原因である環境の変化に対応した情報システム構築に資することも含むケーパビリティを定義し、ケーパビリティ

の下、各システムのシステム化計画が策定される。ケーパビリティについて、アメリカ国防省[8]の例を付録 A に挙げる。情報システムよりより上位のシステムの話である。

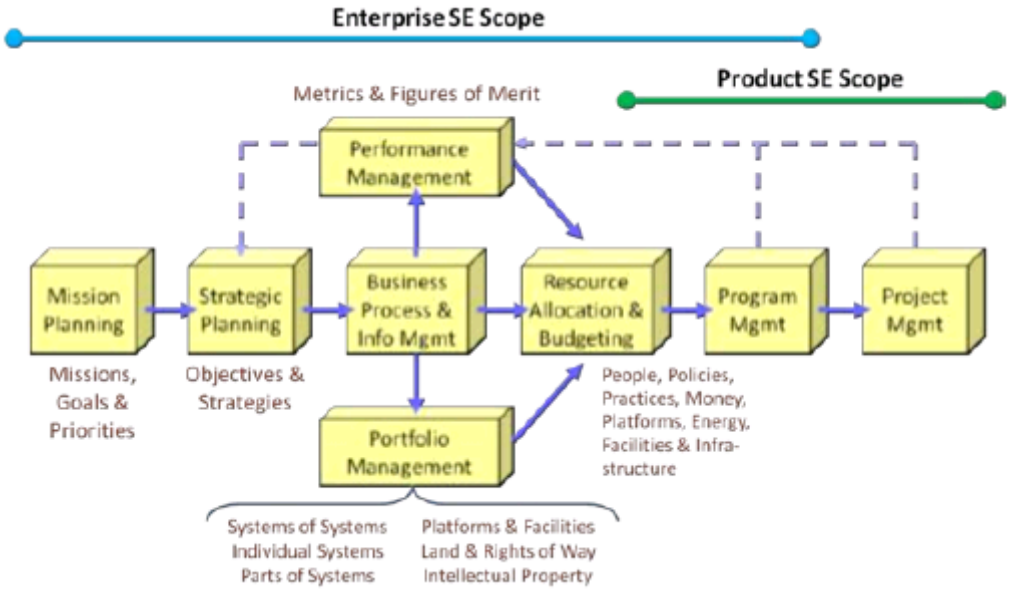
また通常の変動についてはケーパビリティ定義の範疇での要求の変化となるが、想定外のようなケーパビリティを定義していない範囲の要求の変更が生ずる場合は、outer loop の機能としてケーパビリティ自身の改善サイクルを働かせ、各システムの改善も図る。図 9 に既存システムズエンジニアリング（Traditional Systems Engineering）プロセスやビジネスプロセスとの関係性を示した。ビジネスマネジメントとシステム構築の間、システム化計画を拡張したイメージでのプロセスとして位置づけられている。また、ESE をビジネスマネジメントの支援という視点からまとめたのが、図 10 である。ポートフォリオマネジメントなど事業におけるシステム全域が対象となっている。なお、既存システムズエンジニアリングプロセスは、各自活用しているシステム開発プロセスと置き換えても同様に考えても問題ない。



文献[1]Figure1.4 を基に作成

図 9　ESE と既存開発との関係性例

8 システムズエンジニアリングは防衛関係での活用が多く、例の多くが防衛関係となる。



出典：文献[4]

図 10　関係アクティビティ

(1-2) 技術項目

本項での技術項目は次の 7 項目である。

- システムシンキング
- ケーパビリティに基づく開発の分析
- エンタプライズ有効性評価
- 戦略的技術計画
- 技術および標準化動向調査
- ステークホルダー分析
- ConOps の分析・定義

以下、説明する。

(1-2-1) システムシンキング

システムズアプローチのための思考法である。ものごとを多面的な分析をするための考え方であり、多面性を表現する各観点から鳥瞰した内容の表現力、つまりモデリング能力と考えて良い。要求の変化もモデルを通して考えていかないと影響範囲の分析が難しくなる。

他方、MIT の Nancy Leveson 教授の分析によると統合システムの安全性、すなわち単体システムでは全く問題がないものに結合したシステム（System of Systems）にすることで、安全性を損なうケースがあることがわかっており（IPA/SEC の 2010 年度の

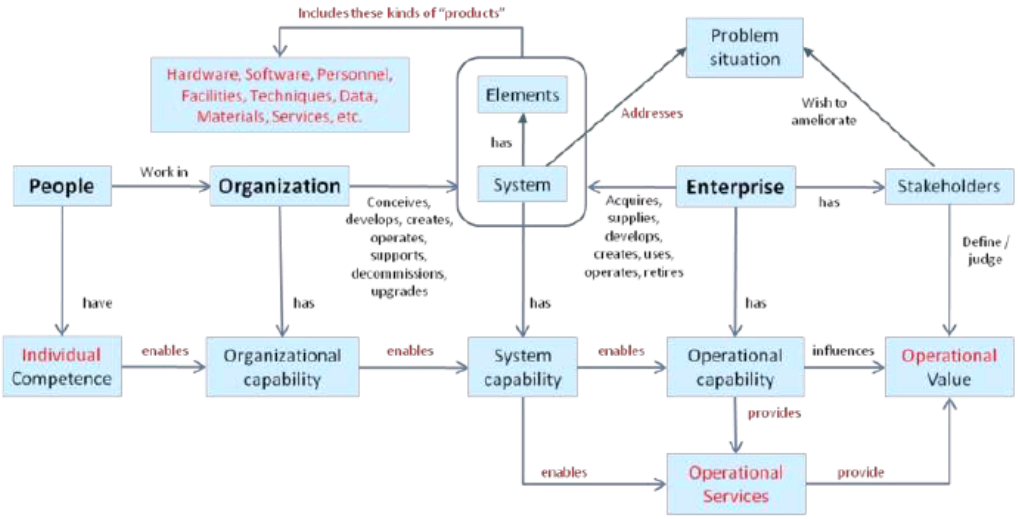
モデリング技術応用 WG においても検討されていた）、問題解決のために必要な技術としてシステムシンキングが挙げられている。そこで、技術項目としては異論があるかもしれないが、システムシンキングはシステムズエンジニアリング分野の全域にも求められている最重要項目であるため、敢えてここで挙げることにした。

(1-2-2) ケーパビリティ[9]に基づく開発の分析（Capability based plannning analysis）

対象組織の目標達成に向けた、対象組織の情報システム群の高水準要求分析のことである。ニーズ、ケーパビリティ、目標達成可能性の認識と評価を実施、アーキテクチャモデリングと評価、統合的なポートフォリオマネジメント、同様に改善計画と実施、それに要求の変化に対応したケーパビリティの修正マネジメントを行う。ステークホルダー要求の変化に対応した開発を許すことができるシステムを導入するために、そのシステムに必要とする能力を事前に把握するために全体プログラムの分析を行うと言える。図 11 に示すとおり次の 3 つのケーパビリティがあり、それらを分析する。

- システムケーパビリティ（system capability）
  システムを実現する直接的な能力のことである。これを欠くとシステム構築に不安が残る。また要求の変化に対応したシステム構築能力も必要となる。
- オペレーショナルケーパビリティ（operational capability）
  ビジネス運用も含めた利用・運用に関する能力のことである。組織としてのサービスを実施する能力となる。システムを使ってビジネスを成功させる能力だけでなく、変化に対応する能力、要求逸脱への対応能力も求められる。
- 組織ケーパビリティ（organizational capability）
  上述を備えた組織能力のことである。組織能力は組織の構成員がシステムを実現そして利用・運用し、ビジネス的な成功を収めるためのコンピテンシーを必要としている。組織ケーパビリティに欠けると様々な問題を発生すること可能性がある。情報システムが「動かない」原因のひとつでもある。

9 「能力」と訳す場合もあるが、日本語での「能力」という言葉の英語のニュアンスが Capability とは異なることから、ケーパビリティと訳す。軍事関係の例を付録に掲載した。



出典：　文献[4]

図 11　3 つのケーパビリティの関係例

(1-2-3) エンタプライズ有効性評価（Enterprise Evaluation and Assessment）

エンタプライズシステムを含むシステム全体の有効性を評価する。システム全体のもつべき全体の統合の在り方を分析し、方向性を決定する。

ここでは、組織ビジョンの実現に向けた進捗度を測定する。測定プログラムを確立し、意図通りに戦略や実装が働いているかを計測する。リスクや目標の達成度、課題の診断等を行う他、感度分析では組織としての要求の変化への対応に関する堅牢性とアジリティの度合いを確認する。シミュレーションを含む分析を実施し、その結果を早めにシステムやビジネスに反映する。

(1-2-4) 戦略的技術計画（Strategic Technical Planning）

エンタプライズシステムが準拠すべき標準やパターンの最小セットを制定する。リスクを避け、目標達成可能性を向上させる技術戦略を策定する。戦略的な技術的計画の目的は、情報システム群を活用する組織全体の技術戦略を確立することである。組織の構成メンバを考慮しつつ、標準的な技術利用と新技術利用のバランスをとる。このために技術と標準化調査で策定したロードマップを活用する。

このロードマップに組織のもつケーパビリティをマッピングし、整合性と相乗性を確認する。整合性と相乗効果がとれないことは、技術戦略上の回避すべきリスクや追求する可能性との調整を必要とする。結果は、プログラムやプロジェクトの実装ガイドラインの観点でまとめられ、活用される。

(1-2-5) 技術および標準化動向調査（Technology and Standards Planning）

環境からエンタプライズに導入すべき技術の兆候を調査する。標準的技術や新技術を含む今後の技術動向および標準化動向を調査し、組織としてのロードマップを策定する。要求の変化に対応するための情報を提供する。

(1-2-6) ステークホルダー分析（Stakeholder Analysis）

ビジネスプロセスと開発プロセスに登場するステークホルダー間の公平性を評価する。当該システムのステークホルダーが誰であり、どのような役割をもっているのかを分析する。要求を変化させる原因となるメンバを明確にする必要がある。

(1-2-7) ConOps の分析・定義

システム全体としての利用・運用の在り方と、それに従ったシステムのコンポーネントの利用・運用の在り方、相互依存関係などを分析して、システムの利用・運用の概念をまとめる。企業や事業レベルでのビジネスレベルで策定されたものを ConOps と呼び、特定の情報システム（System of Systems を含む場合もある）についてを OpsCon（Operational Concept)と呼ぶことがある。成果物については様々な呼ばれ方をするが、概念的な要件を定義したドキュメントであり、ビジネス（運用も含む）で活用するシステムの要件をまとめるためには、非常に重要な書類である。要求の変化への対応も ConOps、OpsCon を活用する。

(1-3) 課題解決効果

システムシンキングは、非常に重要な概念であり、多面的な視点設定を行い、分析することで、対象となるシステム像を洗い出す。特にモデルを用いる開発技術のためには、この技術を習得する必要がある。

それ以外の技術項目は、全項目を挙げて、超上流工程としてエンタプライズシステムズエンジニアリングによる解決を目指す。特に(1-2-2)から(1-2-6)については、文献[1]において、著者らが有効であるとして選択した項目である。宇宙・防衛関連では実績がある手法であり、新しいシステム構築技法への道を開くことが可能となる。

(1-4) 課題解決に向けた留意点・課題

防衛関係では実績や分析が進んでいるが、民生分野では実績があるとはいえない。特にコスト面を下げるためのテーラリングが必要である。他方、一番の懸案事項は、この分野を担える人材が不足していることである。我が国の場合、コンポーネント開発は超一流と言われているが、一方でシステムシンキングができる人材が文系・理系を問わず求められる。

(1-5) 情報のリソース

- INCOSE: Systems Engineering - Systems Engineering Handbook, 2010.
- George Rebovich, Jr. and Brian E. White: ENTERPRISE SYSTEMS ENGINEERING Advances in the Theory and Practice, CRC Press, 2010.
- Nancy Leveson: Engineering a Safer World Systems Thinking Applied to Safety, The MIT Press, 2011.
- http://www.sebokwiki.org/index.php/Guide_to_the_Systems_Engineering_Body_of_Knowledge_%28SEBoK%29_v._0.5
  （The Guide to the Systems　Engineering Body of Knowledge （SEBOK） Version 0.5）
- ISO/IEC/IEEE29148:2011 Systems and software engineering – Life cycle processes – Requirements engineering.

(2) 上流工程における横断的連携

(2-1) 概要

システムズエンジニアリングは、方法論、プラクティスとして非常に詳細なところまで定まっており、ツールによる支援が望まれる分野でもある。要求管理など、ツール化が進んでいるところもあるが、一貫した開発の環境も見られるようになった。構築に関係する部分だけではあるが、それらの方法論となる技術を掲載した。

(2-2) 技術項目

(2-2-1) モデリング・シミュレーション・プロトタイピング（Modeling、Simulation、Prototyping）

システムズエンジニアリングでは、システムをモデル、設計、要求の三要素で特徴づける。モデリングはシステムの振舞や機能性、物理的な状況など特徴を表現するための道具立てである。また数学モデルまで拡張し、実際にコンピュータ上等でモデルを動かして評価するのがシミュレーションである。最近は HILS （Hardware In the Loop Simulation）、SILS（Software In the Loop Simulation）、MILS （Model In the Loop Simulation）、OILS(Operator In the Loop Simulation)など多様なシミュレーション環境が活用される。他方、モデル段階が多いが、限定された仕様で実現してみることをプロトタイピングと呼ばれるが、前者 2 つと相まって、目標とするシステムの分析や実現のベースに用いられる。

システムズエンジニアリングに限らず、ソフトウェアエンジニアリングでも重要な技術である。変化した要求への対応について上流工程段階で設計検証することは非常に重要である。基本的な手法として位置づける。

(2-2-2) 機能ベースの SE 手法活用（Functions-Based Systems Engineering Method）

機能要求の分析と洗い出しを行う。機能要求、性能要求、計画決定要求、仕様および標準化要求、アーキテクチャ概念や ConOps さらに制約などの情報を基に、振舞、コンテキスト、コントロールフロー、データフロー、データディクショナリ、ＥＲ、Functional Flow Block、Models、Simulation Results、Integrated Definition for Functional Modeling の各図を得ることができる。

(2-2-3) オブジェクト指向 SE 手法活用（Object-Oriented Systems Engineering Method）

システムズエンジニアリングのプロセスを OMG（Object Management Group）が中心となってアーキテクチャ記述言語である SysML で記述する方法論を用意している。リアルタイム系の場合 リアルタイムアーキテクチャ記述言語である Marte も活用することがある。なお、アシュアランスについても記述する方法論の研究も進んでいる。

関係アクティビティとしては次のとおり。

- Analyze Needs,
- Define Systems Requirements,
- Define Logical Architecture
- Synthesize Allocated Architectures
- Optimize and Evaluation Alternatives
- Validate and Verify System

(2-3) 課題解決効果

ツール化が可能なため、早期開発およびその保守が効率的に実施できる。要求の変化のビジネス面も含む影響度分析から構築まで、要求の変化に追従したシステムの実現に大きく寄与できる。

(2-4) 課題解決に向けた留意点・課題

ビジネス-IT オペレーションという観点が不足しており、モノを作るという考え方である。モニタリングなどをもっと前面に出す必要がある。

(2-5) 情報のリソース

- INCOSE: Systems Engineering - Systems Engineering Handbook, 2010.

3.1.2.検証技術（高信頼システム）

要求の変化に対応する情報システム構築であっても、高信頼化のためには検証（確認も含む）が欠かせないと考えられる。そのために、調査方法で説明したとおり、お二人の WG 委員も兼ねる有識者の方よりヒアリングを行った。

まず基本的な話として、検証技術についてはこのような開発であったからと言って特段変わるということはないであろうということである。通常開発でも使われている技術を適用することになる。そこで、ヒアリング内容より当該システム構築にとって有効な技術項目を抽出し、それを基に技術課題をまとめた。

なお、技術課題としては本報告書では取り上げていないが、WG での検討（IBM が提供しているアーキテクチャパターン「patterns for e-business」の解説にもビジネス機会について言及がある）にて、次の技術課題が検証（確認）技術として挙げられている。本項目については、技術項目の分析のところで再度取り上げる。

> **コラム　確認なしでのリリース**
>
> Facebook（http://www.facebook.com/）等で適用されるツールは即効性的なリリースが求められている代わりに、リスクを回避できることが前提となっているはずである。同様に e-business では、サービス提供が遅くなることで、顧客名簿の獲得をし損なうリスクの方をケアすることで、できるだけ早くリリースし、ビジネス機会を喪失しないことが重要視される場合がある。そこで、ひとつの対応としてフロントエンドについてできるだけ精度の高い開発を実施する代わりに確認抜きでリリースする。ただしアーキテクチャとして基幹系と直結せずに人間系を通して結合することでリスクを低減させるやり方も考えられる。

(1) 上流工程開発力強化

(1-1) 概要

テスト項目作りは、要件定義書や、仕様書などをベースに作られることから、上流工程成果物の評価手法のひとつとしても考えることができる。検証できない要求や、多くのテスト項目を組み合わせないとテスト項目が作れない仕様書などをみつけるために良いとされている。要求の変化に合わせて早期の短期的な開発を、テスト技法により実現する。

(1-2) 技術項目

(1-2-1) 状態遷移系テスト

状態遷移が要求分析で定義できている場合、要求の変化の影響により状態遷移に対応することで、全体システムで支障がないことを確認することは障害発生の抑止になる。また状態遷移は要求逸脱という動的な変化に対する、障害発生抑止に上流工程において有用である。

(1-2-2) W 字型開発

一般の V 字型開発ではテスト設計が後段にくるが、テスト設計を開発前段に移すことで、品質リスクを軽減する。例えば次の効果が考えられる。

- テストという別の側面から要求を確認することで、要件の矛盾点や抜け・漏れを防止できる可能性がある。
- 要件とテスト項目（テスト技法）の間でのトレーサビリティが確保できる。
- テストの困難さを上流工程で把握することで、良質な設計内容の実現や、テスト作業の早期化ができる可能性がある。

(1-3) 課題解決効果

変化した要求を上流工程の早いうちに問題性を除去しておくことは非常に重要なことである。検知活動ではあるが、予防対策として非常に効果があると考えらる。

(1-4) 課題解決に向けた留意点・課題

人材的に要求を作る技術者とテスト項目を作る技術者が重なることが少ない、つまり両方のスキルをもっている技術者が稀有であることが課題である。

(1-5) 情報のリソース

- 有識者ヒアリング
- IPA/SEC 編： 高信頼化ソフトウェアのための開発手法ガイドブック，2011.

(2) ロバストネスの確保

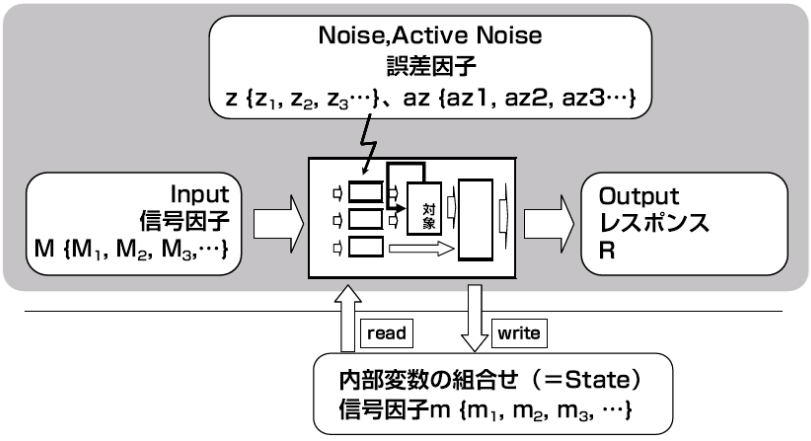
(2-1) 概要

要求の変化や要求逸脱に強いシステムはある一定のロバストネス（堅牢性）が確保されているシステムと言われる。ハードウェアを含むシステムではロバストネスの大小など重要項目となる。当該開発では変更に強い、別の言い方をすればノイズ成分がある入力に強いコンポーネントを実現するための検証技術に注目した。

(2-2) 技術項目

(2-2-1) ロバストネスの確保のためのテスト技法

環境の違い等の少々の変化による改修を避けるために、堅牢性を確保したソフトウェアの実現を確認するロバストネスについてのテストを実施する。テスト工程に合せて組み合わせるべき入力因子について、市場条件となる要因を誤差因子として積極的に取り込んだテストのことである。誤差因子という負荷をかけたテスト（図 12 参照）で、様々な条件下で安定して動作するロバストネスを確保することができる。



出典：IPA/SEC 編：高信頼化ソフトウェアのための開発手法ガイドブック, 2011.([8])

図 12　テストモデル（ラルフチャート）

(2-3) 課題解決効果

ソフトウェアであっても場面によってはロバストネスを確保することが重要となることがある。要求逸脱に対するアシュアランスケースによる対応にも影響がある考え方である。

(2-4) 課題解決に向けた留意点・課題

情報システム開発での適用箇所についての知見を収集する必要がある。

(2-5) 情報のリソース

- 有識者ヒアリング
- IPA/SEC 編： 高信頼化ソフトウェアのための開発手法ガイドブック

3.1.3.非ウォーターフォール型開発技術

非ウォーターフォール型開発技術は、ラピッド型開発や、要求の調整等、要求の変化に対応する情報システム開発に活用可能な技術である。要求の調整についてはプロセスと人により実施されると考えられるが、基本的な技術課題に加え、ラピッド型開発を中心に技術課題をまとめた。網羅的に技術項目とプラクティスを挙げた上で、プラクティスは技術項目として取り上げづらいものが多く、それらを一括して外し、技術項目だけ残した結果である。

ここでは技術項目を整理した結果として、次の技術課題を得た。

- 非ウォーターフォール型開発技術のための要求の明確化
- 非ウォーターフォール型開発技術のための構成管理
- 短期間での開発
- 反復による開発
- 非ウォーターフォール型開発技術のための品質管理

一般的な技術課題と差別化するために「非ウォーターフォール型開発技術のための」という修飾語を付加した。

(1) 非ウォーターフォール型開発における要求の明確化

(1-1) 概要

顧客のニーズに従った商品を適時に提供するため、システムを絶えず進化させるための前提として要求の変化を明確化する必要がある。そのため、要求を明確化する技術を必要とする。

要求の明確化には、機能仕様・パフォーマンス仕様の定義や、ビジネス要件やシステムがサポートするビジネスプロセスの範囲を明確化した上でのプロトタイプの作成などが技術課題として挙げられる。

(1-2) 技術項目

(1-2-1) 機能仕様の定義

非ウォーターフォール型開発手法では、少なくとも次のイテレーションで実現する機能については明確に定義する必要がある。Planguage を使って機能要求仕様を記述することも考えられる。

(1-2-2) パフォーマンス仕様の定義

システムパフォーマンスとは、システムがどの程度うまく動くか、どんな利益があるか、環境にどう影響するか、などである。パフォーマンス仕様をインクリメンタルに記述し、改良していくことが必要である。パフォーマンス属性は、具体的には次の 3 種類に分類される。

- 品質 – 信頼性、使いやすさなど
- 作業能力
- リソース節約

(1-2-3) プロトタイピング

要求の明確化を行うには、プロトタイピングが有効である。プロトタイプを多用する DSDM（Dynamic System Development Method）では、システムの評価を行うためのビジネス・プロトタイプ、ユーザビリティ・プロトタイプ、性能/容量プロトタイプ、機能/設計プロトタイプなどが用いられる。

(1-2-4) 実現可能性調査

ビジネス要件を技術的に実現可能かどうか確認し、技術的な解決方法の候補を洗い出すとともに、大まかな時間とコストの見積もりを行う。ビジネス分析者、エンドユーザ、技術者、ユーザ側の上級管理者で行う必要がある。成果物としては、計画概要書などが挙げられる。

(1-2-5) ビジネス調査

システムがサポートするビジネスプロセスの範囲を確定し、要件に優先順位を付けることが重要である。今後の開発の技術的な基盤をさらに明確にし、非機能的な要件を確定するための調査でもある。成果物としては、ビジネス分野定義書、優先度の付いた要件の一覧、システムアーキテクチャ定義書、プロトタイプ概要計画書などが挙げられる。

(1-3) 課題解決効果

機能仕様の定義やパフォーマンス仕様の定義を実施することで要求を明確化していくことが可能である。また、DSDM のような開発手法では、要求の明確化にプロトタイプが多用される。実現可能性調査やビジネス調査を実施することで抽象度の高い処理要件と情報要件を識別し、その要件に基づいて、プロトタイプを構築しコミュニケーションを密に行いながら設計・構築を進める。

(1-4) 課題解決に向けた留意点・課題

非ウォーターフォール型開発手法は、要求が日々変化することを前提とした手法であり、ここで示した技術課題だけでなく、イテレーションなどの開発手法全体を通して、イテレーション毎の要求を明確化していることに注意が必要である。

(1-5) 情報のリソース

- Tom Gilb, Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage, (Butterworth-Heinemann, 2005)
- DSDM CONSORTIUM, DSDM Consortium - Enabling Business Agility, http://www.dsdm.org/

(2) 非ウォーターフォール型開発における構成管理

(2-1) 概要

短期間の開発サイクルを回す非ウォーターフォール型開発手法では、日々変化する要求の変化に対応するために、開発工程における多くのマネジメント技術や開発技術がある。各ステークホルダー要求の変更に対応し、頻繁に修正を入れつつ管理を行うためには、コーディング規約や所有権の管理、トレーサビリティ、ビルド自動化や自動テストといった継続的な統合にかかわる技術などが重要となる。

(2-2) 技術項目

(2-2-1) コーディング規約

障害発生の抑制に応えるために、コードの品質を一定に保つ必要があり、コーディング規約を導入する必要がある。

(2-2-2) 所有権管理

短期間の開発サイクルをチームで回すためには、チーム全体が共同で全てのコードに責任を持つ必要があり、同時に各クラスに関しても責任を持つプログラマを決める必要がある。所有者は個人かサブチームかチーム全員かなどが考えられるが、明確に定めておくことが重要である。

(2-2-3) トレーサビリティ管理

全ての製品（ドキュメント、ソフトウェア、テストコードなど）の進化を管理するため、全ての製品の変更を保存しておかなければならない。プロジェクトライフサイクル中に行われた変更はすべて元に戻せるようにしておく必要がある。

(2-2-4) 継続的統合

非ウォーターフォール型開発手法では、継続的な統合が重要である。Scrum では、少なくとも 1 日 1 回、プロジェクトのチェックインされたコードをすべて統合することが求められている。継続的統合には、ビルド自動化や自動テストなどのツールの活用も必須である。

(2-3) 課題解決効果

短期間の開発サイクルを回すためには、コードを共有することが必須であり、所有権の管理が重要である。クラス毎の所有権の管理、作業成果物の所有権の管理、さらにはユーザへの引渡し権限なども管理することにより、開発速度が向上する。また、コードを共同所有する中で、コーディング規約、トレーサビリティに加え、ビルド自動化や自動テストにより継続して統合することがコードのデグレードの防止に繋がる。

(2-4) 課題解決に向けた留意点・課題

要求の変化に対応するために、顧客に常に見せることができる状態にしておくという意味でも継続的な統合はメリットがある。構成管理自体は非ウォーターフォール型開発手法にかかわらず、重要なことではあるが、継続的な統合のための技術は同開発手法では特に重要である。

(2-5) 情報のリソース

- Craig Larman, Agile and Iterative Development: A Manager's Guide, (Addison-Wesley Professional, 2003) [訳: 児高慎治郎ら, 初めてのアジャイル開発 ~スクラム、XP、UP、Evo で学ぶ反復型開発の進め方~, (日経 BP 社, 2004)]
- Prentice Halltephen R Palmer, John Mac Felsing, A Practical Guide to Feature-Driven Development, (Prentice Hall, 2002)
- Alistair Cockburn, Crystal Clear: A Human-Powered Methodology for Small Teams, (Addison-Wesley Professional, 2004)
- DSDM CONSORTIUM, DSDM Consortium - Enabling Business Agility, http://www.dsdm.org/

(3) 短期間での開発

(3-1) 概要

非ウォーターフォール型開発手法では日々変化する要求の変化に対応するために、短い開発工程を繰り返し頻繁にリリースし、ステークホルダーとともに評価と計画の見直しを実施する。短い開発工程を実現するための技術として、コード自動生成やシミュレーション、テスト駆動開発などが挙げられる。

(3-2) 技術項目

(3-2-1) 機能毎の開発

非ウォーターフォール型開発手法では既存のソフトウェアコンポーネントを活用することにより短期間での開発を実現する。そのためにも機能毎に開発する必要があり、FDD（Feature Driven Development）では 2 週間以内に実装できない機能はより小さな部分機能に分割される。

(3-2-2) コード自動生成

短期での開発には、自動コード生成アプリケーションの利用も考えられる。

(3-2-3) シミュレーション

変更内容を複数の開発関係者で素早く確認するために、シミュレーションを活用する。モデリングや影響評価が手法として挙げられる。

(3-2-4) テスト駆動開発

テストを書き、そのテストをパスするように目的のコードを記述するというテスト駆動開発は、短期間で実装と検証を繰り返すものでもあり、短期間での開発に大きなメリットがある。

(3-2-5) シンプルな設計のための技術

シンプルな設計は短期の開発には必須であり、そのための技術も多い。システムのメタファやテスト測定基準の要求仕様への記述、オープンエンドのアーキテクチャ、実行可能なスケルトンの構築、インクリメンタルなアーキテクチャの再設計などが挙げられる。

(3-3) 課題解決効果

機能毎の開発やコード自動生成、シンプルな設計等の技術を活用することにより、開発速度が向上し、短期間での開発が可能となる。

(3-4) 課題解決に向けた留意点・課題

短期間での開発に必要な機能毎の開発やシンプルな設計のための技術には、非ウォーターフォール型開発手法でも多くの手法があり、開発案件に応じて適切な手法を選択する必要がある。

(3-5) 情報のリソース

- Prentice Halltephen R Palmer, John Mac Felsing, A Practical Guide to Feature-Driven Development, (Prentice Hall, 2002)
- Roger S. Pressman, Software Engineering: A Practitioner's Approach, (McGraw Hill Higher Education, 2004) [訳: 西康晴ら, 実践ソフトウェアエンジニアリング -ソフトウェアプロフェッショナルのための基本知識-, (日科技連出版社, 2005)]
- Tom Gilb, Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage, (Butterworth-Heinemann, 2005)
- Craig Larman, Agile and Iterative Development: A Manager's Guide, (Addison-Wesley Professional, 2003) [訳: 児高慎治郎ら, 初めてのアジャイル開発 ~スクラム、XP、UP、Evo で学ぶ反復型開発の進め方~, (日経 BP 社, 2004)]
- Alistair Cockburn, Crystal Clear: A Human-Powered Methodology for Small Teams, (Addison-Wesley Professional, 2004)

(4) 反復による開発

(4-1) 概要

非ウォーターフォール型開発手法では開発サイクルが繰り返されるため、繰り返しのサイクル（イテレーション）を管理しつつ、ステークホルダーからの要求を適切に取り込んでいくための技術を必要とする。繰り返しの中で要求を取り込むためには、ユーザからのフィードバックを含めたスプリントレビューやサイクル開始前のスプリント計画が重要であり、プロジェクトの管理にはリファクタリングや、タイムボックス等を利用した頻繁なリリースの管理が挙げられる。

(4-2) 技術項目

(4-2-1) リファクタリング

リファクタリングは反復による開発では重要であり、全てのテストが通ることを確認しながら、きめ細かいコードや大規模な設計要素を単純化していく必要がある。

(4-2-2) イテレーション計画

イテレーション計画を立てることで、イテレーションの度に要求を明確化していく。利害関係者があつまって、優先順位を付け直し、今回のイテレーションの目標を決定するタスクと、チームとプロダクトオーナのみで集まって要求をどう実現するか熟考し、目標を達成するためのタスクを列挙するタスクなどが含まれる。

(4-2-3) スプリントレビュー

各イテレーションの最後に、レビューミーティングを行うことで要求を明確化し、イテレーションの意義を高める。チーム、プロダクトオーナ、その他の利害関係者により実施し、システムの機能、設計、長所/短所、チームの作業、今後問題が発生しそうな箇所について明確にする。

(4-2-4) 頻繁なリリース管理

反復による開発における頻繁なリリースのための管理技術としては、スプリントバックロググラフの作成、スダンドアップミーティング、バーンダウンチャートによる可視化、タイムボックスなどが挙げられる。

(4-3) 課題解決効果

リファクタリングをし、設計やコードを単純化しつつ、イテレーション計画やスプリントレビューを実施することで、繰り返しのサイクルを管理しつつ、ステークホルダーからの要求を適切に取り込んでいくことが可能となる。

(4-4) 課題解決に向けた留意点・課題

頻繁なリリースの管理には、多くの可視化手法やタイムボックス等のプラクティスがあるものの、活用には熟練した技術を必要とするものが多い。

(4-5) 情報のリソース

- Craig Larman, Agile and Iterative Development: A Manager's Guide, (Addison-Wesley Professional, 2003) [訳: 児高慎治郎ら, 初めてのアジャイル開発 ~スクラム、XP、UP、Evo で学ぶ反復型開発の進め方~, (日経 BP 社, 2004)]
- James A. Highsmith, Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, (Dorset House, 1999) [訳: 山岸 耕二ら, 適応型ソフトウエア開発-変化とスピードに挑むプロジェクトマネージメント, (翔泳社, 2003)]"
- DSDM CONSORTIUM, DSDM Consortium - Enabling Business Agility, http://www.dsdm.org/
- Alistair Cockburn, Crystal Clear: A Human-Powered Methodology for Small Teams, (Addison-Wesley Professional, 2004)

(5) 非ウォーターフォール型開発における品質管理

(5-1) 概要

非ウォーターフォール型開発手法では、早期開発による品質の低下、人に依存した品質のばらつきを防ぐための技術が必要であり、ソフトウェアインスペクションやペアプログラミングといった技術が利用されている。

(5-2) 技術項目

(5-2-1) ソフトウェアインスペクション

インスペクションにより、主にバグを検出することで設計やコードの品質を保証する。また、欠陥を発見するとともに、チームとしてのコラボレーション能力を高めるメリットもある。

(5-2-2) ペアプログラミング

ペアプログラミングは、エクストリームプログラミング（XP）のような手法では核となるものであり、コードは必ず 2 人のプログラマが 1 台のコンピュータに向かって作成し、定期的に交代しながら交互にコーディングする。常にレビューしている状態になり、品質の向上にはメリットがある一方で、ペアの相性や技術者のスキルなどが要求される。

(5-3) 課題解決効果

インスペクションによって欠陥を発見するとともに、チームとしてのコラボレーション能力を高めることが重要である。ペアプログラミングはエクストリームプログラミング（XP）などで多く用いられる手法であり、品質向上に加え、開発効率の向上や教育効果なども期待される。

(5-4) 課題解決に向けた留意点・課題

ペアプログラミングは、開発チームの技術者に高いスキルを要求することやペアの相性の問題などがあり、活用には注意が必要である。

(5-5) 情報のリソース

- Prentice Halltephen R Palmer, John Mac Felsing, A Practical Guide to Feature-Driven Development, (Prentice Hall, 2002)
- James A. Highsmith, Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, (Dorset House, 1999) [訳: 山岸 耕二ら, 適応型ソフトウエア開発-変化とスピードに挑むプロジェクトマネージメント, (翔泳社, 2003)]

- Tom Gilb, Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage, (Butterworth-Heinemann, 2005)"
- Craig Larman, Agile and Iterative Development: A Manager's Guide, (Addison-Wesley Professional, 2003) [訳: 児高慎治郎ら, 初めてのアジャイル開発 ~スクラム、XP、UP、Evo で学ぶ反復型開発の進め方~, (日経 BP 社, 2004)]
- Alistair Cockburn, Crystal Clear: A Human-Powered Methodology for Small Teams, (Addison-Wesley Professional, 2004)"

3.2. WG の議論による追加技術

WG において種々の技術について議論を実施し、そのなかで運用技術について ITIL を追加することとした。要求の変化に対応する情報システムは継続性が求められるため、開発だけではなく利用・運用時の対応も重要視する必要がある。運用技術のベストプラクティスとして最も有名な技術のひとつである ITIL を調査する。特に ITIL V3 からはビジネス面とのつながりが強調されており、超上流工程の技術としても注目した。

(1) 運用技術

(1-1) 概要

要求の変化に対応するシステム構築技術は、要求管理者の下、継続的にシステムが開発されるイメージであり、その前提が IT を使ったサービス（ここでは IT サービスと呼ぶ）の運営が実施されていることである。ビジネス活動の一部として IT サービスの運用があり、ビジネスと IT サービスの両方のニーズから情報システムが構築され、要求の変化はビジネスおよび IT サービスの運用に関わるステークホルダーの環境変化の気づきから発生すると考えられる。また要求の変化の結果によって変化した情報システムの受け取り手のひとりが運営先であり、運営面の IT サービスの運用技術（特にマネジメント技術）がシステム開発と比較して前面にくる。

(1-2) 技術項目

(1-2-1) ITIL 継続的改善

ITIL（Information Technology Infrastructure Library）は、IT サービスマネジメントのベストプラクティスをまとめたフレームワークである。詳細は日本の代表機関である itSMF（特定非営利活動法人 IT サービスマネジメントフォーラムジャパン）のホームページ[10]を参照されたいが、組織全体に展開するメリットとして、

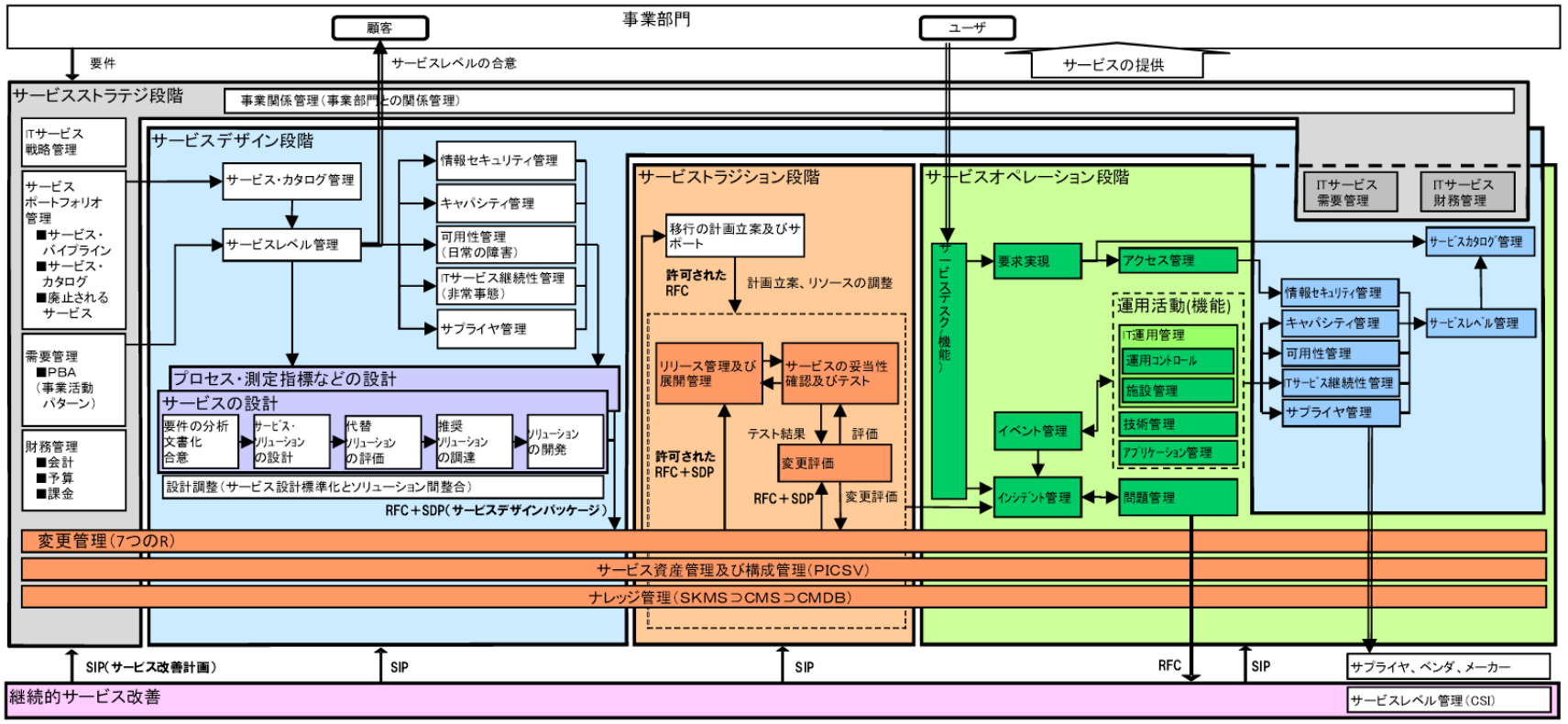
- IT サービスに対するユーザと顧客の満足度の向上
- サービス可用性の向上、事業の利益と収益の増加
- やり直しと損失時間の削減、リソースの管理と利用の改善によるコスト削減
- 新しい製品やサービスの市場投入までの時間の短縮
- 意思決定の改善とリスクの最適化

が挙げられている。ITIL は第 2 版までは IT システムの運用面に注目されていたが、ITIL 第 3 版ではビジネスとの関わりが協調されており、ビジネス面からの要求の変化、IT

[10] http://www.itsmf-japan.org/itil/

サービス面からの要求の変化の両面性を持っている技術である（図 38、図 13、図 14 を参照）。

図 13 のとおり、ステークホルダー管理を行い、運用でわかった事項を適切なステークホルダーに届けることになる。継続的サービス改善によって、各段階へ SIP（サービス改善計画）が与えられる。事業部門からの要求はサービスストラテジ段階へ送られ、逆にサービスが提供される。

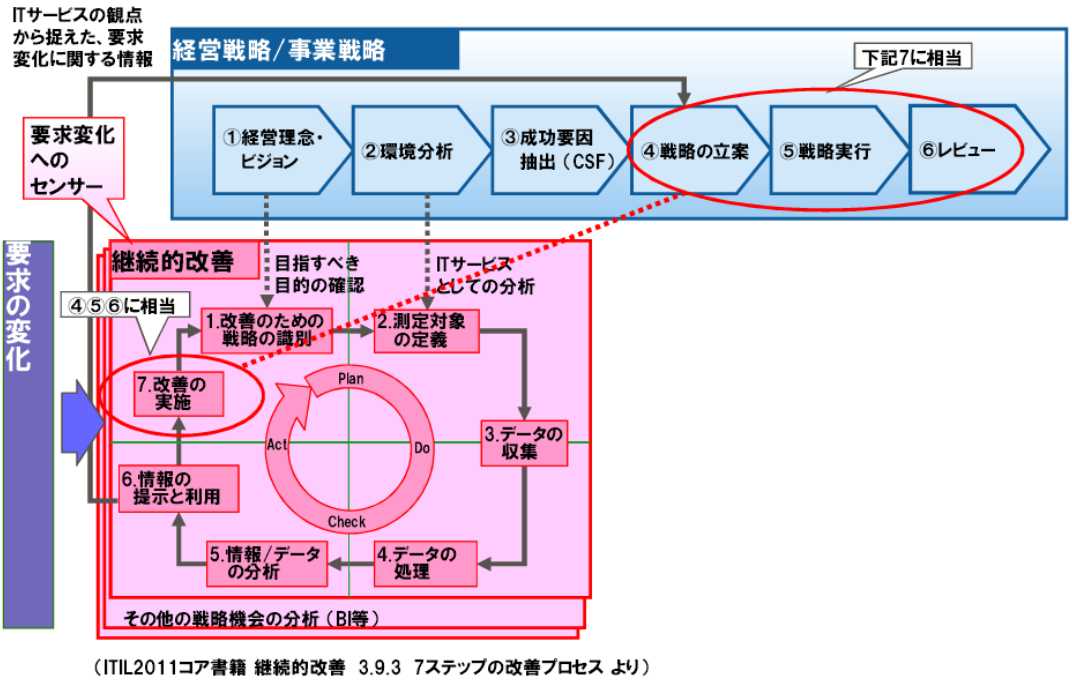


成瀬 WG 委員資料（itSMF Japan コンファレンス富士通講演より）([6])を一部改編

図 13　ITIL V3（2011 年版）の全体プロセス概念

図 14 のとおり、要求の変化に呼応するのが継続的改善であり、その結果が超上流工程のプロセスの入力のひとつになり、運用側からの要求変化にきっかけとなる。7 つの改善プロセスは次のとおり。

- 改善のための戦略の識別
- 測定対象の定義
- データの収集
- データの処理
- 情報/データの分析
- 情報の提示と利用
- 改善の実施



成瀬 WG 委員資料（itSMF Japan コンファレンス富士通講演より）([6])を一部改編

図 14　IT サービスの観点による要求変化への対応

(1-3) 課題解決効果

IT サービスというとどうしてもシステム開発の陰のような存在として見る向きもあるが、本課題では大きく前面に出てビジネスマネジメントと連携して IT システムをコントロールする役割を期待する。

(1-4) 課題解決に向けた留意点・課題

IT サービスは、ビジネス要求の一環として運用要求を策定するが、一方でビジネスと同様にシステム開発側の分析結果の影響から図 37 のとおり、やはりビジネスと同様に運用要求の変更（開発側より詳細な情報の提供を受けることも含む）をする必要な局面も出てくる改善サイクルの存在を理解している必要がある。

また人材的に運用要求の分析と実際の運用実施の両面ができる技術者が少ないことが課題である。

(1-5) 情報のリソース

- WG 資料
- itSMF（特定非営利活動法人 IT サービスマネジメントフォーラムジャパン）のホームページ（http://www.itsmf-japan.org/itil/）

3.3. 2010 年度調査等を加えた再整理結果

3.3.1. 2010 年度調査に基づく技術課題の再整理

2010 年度調査に対して、再整理した結果を本節に記す。2010 年度調査については、内容を次の方針で再整理した。なお、2010 年度調査は、有識者ヒアリングによりすべての技術項目が調査されていた。

- 通常開発でも重要な技術は割愛し、本調査における重要項目を浮き出させた。そのため、例えば「要求獲得」、「検証技術」一般は掲載しなかった。
- 有識者ヒアリング結果まで立ち戻り、2011 年度調査の観点から必要な技術を再度選別した。
- 分類名を本調査の趣旨に沿うよう調整した。それに伴い、一部の技術課題を別の分類に移動させた。

3.3.1.1. 要求管理

要求管理に関する技術課題である。変化した要求を正確に記述するために『要求の明確化』、変化する要求の範囲を定めるために『スコープ決め』、さらに要求トレーサビリティを確保し、一方で将来的な要求も含めた『要求記述』が求められている。

(1) 要求の明確化

(1-1) 概要

要求の変化に対応するためには、『要求の明確化』が求められる。要求の明確化のために、業務担当と仕様記述者の連携、コーディングレベルの仕様記述が必要となる。他方、『ドメイン分析と概念モデリング』は、固定部分と変動部分の要求を明確化するために必要な技術課題である。

(1-2) 技術項目

(1-2-1) GSN

アシュアランスケースは要求の変化のうち要求逸脱をはじめとする様々な場面で有効である。要求の明確化の一環で記法の重要性が説かれている。アシュアランスケースを明確に記述するためには GSN (Goal Structuring Notation)を用いる。GSN はゴール

指向の表記のひとつで、主張、立証、証拠を順々にツリー上に記述する。文脈、仮定、正当化、戦略、選択に関する記述を付加することで、内容が明確になる。

(1-2-2) 形式手法

形式手法には様々な手法があるが、プログラムや機能仕様を、論理系をベースにした記法に従って記述することで、静的検証可能なプログラムや仕様を作成する手法である。

手法の用いる論理系によって一般に用途が異なる。また、論理系の知識やスキルがない普通の技術者が使えるよう、様々な工夫がなされている。

ここでは論理系に基づいた要求に対応する仕様を作成するために重要な技術として挙げられている。

(1-2-3) アーキテクチャ記述

要求仕様から詳細設計を行い、プログラムモジュールを作成するのが、ソフトウェア開発であると考えている技術者は多い。機能仕様や単体ソフトウェア/システムならそれでも可能であるが、実際には複合化した構造のソフトウェア/システムも多い。いわゆる非機能要求を満たすことを考える場合、コンポーネント(最小単位のシステム構造)間の静的構造と動的振舞いを作成して解決することになる。アーキテクチャ記述は、そのための記法であり、複合システムを記述する方法である。要求の変化の方向がわかっている場合は、それに強い構造を定義する必要がある。

(1-3) 課題解決効果

要求仕様が明確になり、下流工程の設計に無理なくつなげることで、手戻りをなくし効率化が図れる。また、作成されたドキュメントの可読性を高めることで、要求仕様作成作業が容易になる。

(1-4) 課題解決に向けた留意点・課題

全体として教育・訓練が必要である。GSN は、それを使って保証することを記述すること自体が困難であり、訓練が必要である。形式手法は、論理系の理解と記述の教育、アーキテクチャ記述はアーキテクチャ記述言語などの手法・技法の普及がまず必要である。

(1-5) 情報のリソース

- 有識者ヒアリング

(2) スコープ決め

(2-1) 概要

スコープ決めは要求変化への対応手段であり、要求変化への対応が要求管理そのものと考え、変動部分を設定することである。

(2-2) 技術項目

(2-2-1) 変動部分の明確化

計画的に要求を変化させることが可能な部分を設定するために、変動部分を明確化する技術である。

(2-2-2) 投資効果と市場と技術変化の見極め

要求の変化への対応に対する費用対効果の見極めをする技術である。

(2-2-3) ドメイン分析と概念モデリング

(2-2-1)の具体的な手法のひとつとして、問題の領域の要求の特徴を分析し、さらに顧客の要求をモデル化する。既存の要求を概念モデリングで整理し、新たな要求との違いを見出すことで、最小限の手間での開発を実施する。

(2-3) 課題解決効果

システムの枠組みや構造化、そして変動部分を明確にすることで、修正範囲を特定でき、修正のための設計作業が明確になる。また、プログラム変更に際しても、変更部分に焦点を絞って素早く修正を行うことができ、開発工程の効率化が図れる。

(2-4) 課題解決に向けた留意点・課題

要求の変化をとらえるためには、ドメインの知識がないと一般的に難しい。ドメイン分析と概念モデリングの考え方が重要な課題である。

(2-5) 情報のリソース

- 有識者ヒアリング

(3) トレーサビリティ管理/ベースライン管理

(3-1) 概要

ビジネス要求とシステム要求、そしてアーキテクチャ、プログラムモジュール、さらにテストケースまでのトレーサビリティが求められている。特に認証の取得が必要なケースはトレーサビリティを求められることが少なくない。非ウォーターフォール型開発のアジャイル型開発のように、比較的小規模なソフトウェア開発に向いた手法を使った、

全体システムの開発では、要求の詳細化が均一でなくなる。そのため、複合化した段階的詳細化の管理をする機能であるベースライン管理は重要な技術である。

(3-2) 技術項目

(3-2-1) RTVM

Requirements and Traceability and Verification Matrix のことであり、要求と検証に至るトレーサビリティを確保するための行列のことである。実際にはこれを行列としではなく、ツールの機能として実現することで大規模なシステムでの利用が可能となっている。

(3-3) 課題解決効果

要求定義書や仕様書、変更履歴、テストや障害の記録、ソースコードなどを相互に関連付けることで、仕様変更や欠陥がどのドキュメントに影響しているかの追跡が容易になり、要求を的確に情報システム構築に反映することができる。この結果、開発の手戻りがなくなり、効率化が図れる。また、要求の変化に対して影響する範囲を特定できるため、テスト範囲も明確になる。

(3-4) 課題解決に向けた留意点・課題

特に、非機能要求に対するトレーサビリティを支援する方法論は十分とは言えない状況である。

(3-5) 情報のリソース

- 有識者ヒアリング

(4) コントロールケースの活用

(4-1) 概要

将来要求がある場合やミスユースケースの発生、さらに SLA が守られなくなった場合の行動パターン、システムの状態により稼働状況を変更する場合などの対応方法を記述することで、要求の変化への対応を行う。

(4-2) 技術項目

(4-2-1) コントロールケース

詳細は文献[9]を参照されたい。想定する要求の変化を記述するためのメタモデル（もしくは表記方法）である。将来要求を盛り込む場合などに有効である。

(4-3) **課題解決効果**

許容する要求の範囲とその際の挙動を、アシュアランスケースを使わずに記述できる。

(4-4) **課題解決に向けた留意点・課題**

一般的な実績が少ないため、実践力を高め、実績を増やす必要がある。

(4-5) **情報のリソース**

・ WG での議論

3.3.1.2. システム構成関連技術（部品化・連携）

要求の変化に強いシステムのアーキテクチャを策定するための技術課題が求められている。

(1) **構造の明確化**

(1-1) **概要**

部品化は構造の明確化が前提となり、またモデルとビューの分離が部品化のための構造の明確化につながる。フレームワークが部品構造を明確化するために重要である。

(1-2) **技術項目**

(1-2-1) **アーキテクチャ分析**

要求の変化の方向について調査を行い、それに耐えうるアーキテクチャを設定できるか分析し、設計に活かす技術である。

(1-2-2) **形式手法**

要求を変化させる場所を特定し易くするために、矛盾点を除去しておきたい。そこで、形式手法を活用することで該当箇所を限定し、構造の明確化する。

(1-2-3) **プロダクトライン**

プロダクトに要求の一部の違いを反映し、効率的な開発を進めるためには構造の明確化が必要となっている。その実現のための技術がプロダクトラインの技術である。

(1-2-4) **プラットフォームアプローチ**

要求の変化する箇所を明確化し、概念モデルを適用し易くするためには部品活用の枠組みであるプラットフォームを活用した開発が必要とされている。ただし、実際には仕様のズレが存在する場合へのマージンを確保していることなど、前述のロバストネスの確保などが重要関連技術となる。

(1-2-5) スケルトンと部品

部品を作成するための仕様の枠組みのことである。想定する品質特性の違いには注意が必要である。

(1-3) 課題解決効果

変化しないところをフレームワークとしてとらえ、変化するところを、修正したり入れ替えたりする単位として部品化することで、保守の範囲を局所化できる。

(1-4) 課題解決に向けた留意点・課題

標準化が一番の課題である。

(1-5) 情報のリソース

・　有識者ヒアリング

(2) 部品化

(2-1) 概要

コンポーネント化がシステムを分解する上で重要であり、部品探索により部品を見つけやすくすることで、部品の活用が容易になる。アーキテクチャの基本パターンとしての部品化はアーキテクチャ設計する際に非常便利である。

(2-2) 技術項目

(2-2-1) SOA

Service Oriented Architecture のことであり、サービス起点での組み立て方法である。部品は良いが、全体を見ることが苦手なところの克服がポイントである。クラウドコンピューティングの一部では、SOA により組み立てる場合がある。

(2-2-2) オブジェクト指向

ひとつのカプセル化の手法であり、部品化の方法として用いられている。
操作方法も同時にカプセル化していることからモジュール性の高い部品化を実現できる。

(2-2-3) 業種・業務特化

実際に利用する場合には、汎用の部品は基本的なものを除いて役立つものが少ない。そのため、業種や業務ごとに部品提供されることが利用頻度の向上につながる。

(2-3) 課題解決効果

既存部品の再利用、SaaS・オープンソースソフトウェアなど外部の部品の活用により、テスト作業を含めた作業期間の短縮ができる。

(2-4) 課題解決に向けた留意点・課題

前述の通り、標準化が課題である。また、非機能要求など品質が異なるものを部品として使うことに伴う品質低下の危険性があるため、ロバストネスを保った部品化が求められる。

(2-5) 情報のリソース

・ 有識者ヒアリング

(3) クラウド活用

(3-1) 概要

クラウドコンピューティングサービスを活用した情報システムの構築を行い、要求の変化に従って、サービスを変更しながら利用することが可能になる。

(3-2) 技術項目

(3-2-1) 性能及び機能のスケーラビリティ

クラウドコンピューティングでは繁忙期や閑散期での要求の違いに対処した効率的な利用を可能とするために、性能と機能についてのスケーラビリティを確保することが重要である。

(3-3) 課題解決効果

システム間、部品間の連携が容易になる。

(3-4) 課題解決に向けた留意点・課題

１社だけでなく、複数社のクラウドコンピューティングサービスや自社のサービスとの連携技術が重要となる。

(3-5) 情報のリソース

・ 有識者ヒアリング

3.3.1.3. 安全と情報セキュリティ

安全性とは、ここでは信頼性・安全性のことであり、システム障害への対応技術のことである。システム同士が次々につながってくる時代であり、そのための安全性を確保するだけでなく、インターネット等を介してシステム連携の隙をついた攻撃に備えるため、情報セキュリティも重要である。最近のシステムの特徴的な技術課題である。

(1) テスト網羅性の保証

(1-1) 概要

回帰テストやテスト自動化などは、テスト網羅性の保証手段として重要である。さらに安全性を確保していることを保証するためにアシュアランスケースを記述することも重要である。

(1-2) 技術項目

(1-2-1) アシュアランスケース

ゴール指向の記法の GSN を用いて、エビデンスから安全性を確保していることを説明するために記述するものである。

(1-2-2) トレーサビリティ

認証関係で仕様通り確実に実装ができていることを示すため、トレーサビリティの結果を提示し、それに対するテスト網羅性を示すことで全体を保証する。

(1-2-3) プロダクトライン

プロダクトラインは安全性を保つために粒度を明確化する際の支援をする。

(1-2-4) 形式手法

段階的詳細化による手法により要求の漏れを減らし、テスト網羅性と相まって全体を保証する。

(1-2-5) テスト網羅性証明

テストケースが充足していることを示す。実験計画法等の手法により余計なテストケースを削減する。

(1-2-6) 回帰テスト

要求の変更に対して、影響がないことを確認するためのテストである。

(1-2-7) テスト自動化

繰り返しテストを実施するために、単体テストについては自動化されていることも多い。

(1-3) 課題解決効果

要求が明確になっていれば、もしくは、影響範囲を特定できるような工夫が設計工程でできていれば、回帰テストを実施する範囲が明確になり、テスト作業を効率化できる。また、要求が明確になっていることで、テストの網羅性や粒度を明確に定めることができ、テスト作業の実効性が高まる。テスト自動化は繰り返し実施されるテスト作業自体の効率化につながる。

(1-4) 課題解決に向けた留意点・課題

認証関係でテスト網羅性を求められることも多くなっているが、一方で情報システムでは膨大なテストを実施することで、保証できると誤解している場合がある。重要な局面についてのテスト網羅性を、効率的に確保することが重要である。

(1-5) 情報のリソース

・　有識者ヒアリング

(2) 不測の管理・仮定の管理

(2-1) 概要

不測の事態の発生に対する対処方法は重要であり、そのために絶えず要求を変化させることは重要である。また現実問題として情報システムに実装されているのはモデルであるため、時間の経過とともに要求の逸脱が大きくなる可能性がある。仮定を管理しておくことは不測の事態への対応のためにも重要である。

(2-2) 技術項目

(2-2-1) アシュアランスケース

次の使い方として有効な技術である。アシュアランスケースを記述し、不測への対応の保証、要求の逸脱への対処などすべてアシュアランスケースの記述と連携したシステム構築・運用が重要である。

(2-3) 課題解決効果

不測の事態の発生時・発生前の対処方法を事前に洗い出すことでシステム障害の発生を抑えるものである。

(2-4) 課題解決に向けた留意点・課題

実際にアシュアランスケースを記述するスキルが問題になる。

(2-5) 情報のリソース

・　有識者ヒアリング

(3) 運用性・可用性・アシュアランスの評価

(3-1) 概要

運用性、可用性、アシュアランスを確保するためにこれらの状況を評価し、それに従って場合によっては、必要な要求を変化させることがある。

(3-2) 技術項目

(3-2-1) アシュアランスケース

逸脱などの状況をモニタリングする仕組みを取り入れて評価し、必要とあれば要求を変化させて対応させる記述を行う。

(3-3) 課題解決効果

モニタリングにより、障害の発生を事前に抑える仕組みは重要である。

(3-4) 課題解決に向けた留意点・課題

アシュアランスケースを記述する能力をもった人材が必要である。

(3-5) 情報のリソース

・　有識者ヒアリング

(4) 共通基盤としての保証

(4-1) 概要

仮想化は共通基盤として重要な技術であり、マルチテナントにより利用者側に共通サービスとしての提供が可能となるが、そのための情報セキュリティを共通基盤として確保することは重要である。

(4-2) 技術項目

(4-2-1) 情報セキュリティ

要求の変化に対して、運用面も含めて、共通基盤として脆弱性対策のなされた基盤を提供することは重要である。

(4-3) 課題解決効果

サービス提供側が、サービスレベル、品質、セキュリティの保証を SLA で明確にすることで、サービス利用側でのテスト確認項目が限定される。

(4-4) 課題解決に向けた留意点・課題

運用面での解決が重要である。基盤だけ情報セキュリティを確保しても、運用面に課題が残ると意味が無い。

(4-5) 情報のリソース

・ 有識者ヒアリング

(5) 他者に求められる保証

(5-1) 概要

情報セキュリティ認証により、情報セキュリティ機能の適切性や確実性を第三者が評価し、認識することができる。検証・監査により各種ガイドラインに沿った運用手続きが行われていることの第三者による評価がなされ、改善見直しにつながる。

(5-2) 技術項目

(5-2-1) SLA

サービス品質を利用側、提供側で合意しておくことで保証ができる部分がある。そのために SLA を取り交わす。

(5-2-2) 情報セキュリティ保証

情報セキュリティに関する認証を活用して情報セキュリティを確保する。

(5-3) 課題解決効果

セキュリティ管理体制の十分性などに関し他者の保証が得られることで、利用者は安心してサービスを利用できる。また、SLA などの対象項目やレベルについて提供サービス種類毎に標準が示されることで、サービス選定時の基準となり、利用者の導入負荷を軽減できる。

(5-4) 課題解決に向けた留意点・課題

認証と現実運用の相違点がないよう確認と見極めが重要である。

(5-5) 情報のリソース

・ 有識者ヒアリング

3.3.2.技術課題の一覧

2011 年度調査と前項の見直した 2010 年度調査結果をまとめ、整理した結果を以下に示す。

さらに検証技術で洗い出した技術課題はいずれも上流工程における横断的連携技術としてもとらえることができるために移動した。

表 1　技術課題一覧

(1) ビジネス IT オペレーションの技術課題

技術課題	備考(技術項目等)
超上流工程分析・評価	システムシンキング、ケーパビリティに基づく開発の分析、エンタプライズ有効性評価、戦略的技術計画、技術および標準化動向調査、ステークホルダ分析、CONOPSの分析/定義

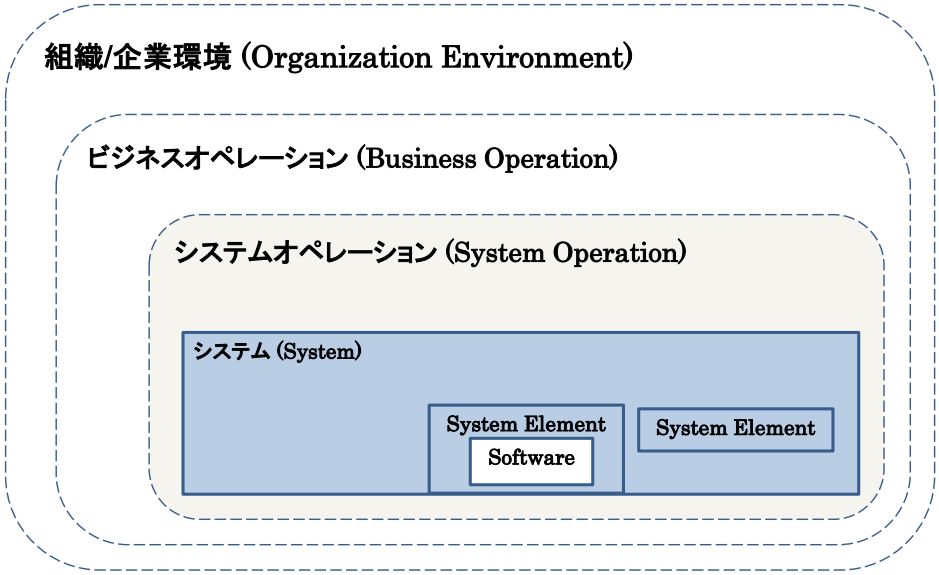
(2) 利用・運用における技術課題

技術課題(大項目)	技術課題(中項目)	備考(技術項目等)
上流工程における横断的連携		モデリング・シミュレーション・プロトタイピング、機能に基づくSE手法活用、オブジェクト指向SE手法活用
	上流工程開発力強化	状態遷移系テスト、W字型開発
	ロバストネスの確保	ロバストネスの確保のためのテスト技法
要求管理	要求の明確化	Goal Structuring Notation、形式手法、アーキテクチャ記述
	スコープ決め	変動部分の明確化、投資効果と市場と技術変化の見極め、ドメイン分析と概念モデリング
	トレーサビリティ管理/ベースライン管理	RTVM
	コントロールケースの活用	コントロールケース
システム構成関連技術(部品化・連携)	構造の明確化	アーキテクチャ分析、形式手法、プロダクトライン、プ
	部品化	SOA、オブジェクト指向、業種・業務特化
	クラウド活用	性能及び機能のスケーラビリティ
セーフティ&セキュリティ	テスト網羅性の保証	アシュアランスケース、トレーサビリティ、プロダクトライン、形式手法、テスト網羅性証明、回帰テスト、テスト自動化
	不測の管理・仮定の管理	アシュアランスケース
	運用性・可用性・アシュアランスの評価	アシュアランスケース
	共通基盤としての保証	情報セキュリティ
	他者に求められる保証	情報セキュリティ保証
非ウォータフォール型開発技術	要求の明確化	機能仕様の定義、パフォーマンス仕様の定義、プロトタイピング、実現可能性調査、ビジネス調査
	構成管理	コーディング規約、所有権管理、トレーサビリティ管理、継続的統合
	短期間での開発	機能毎の開発、コード自動生成、シミュレーション、テスト駆動開発、シンプルな設計のための技術
	反復による開発	リファクタリング、イテレーション計画、スプリントレビュー、頻繁なリリース管理
	品質管理	ソフトウェアインスペクション、ペアプログラミング

(3) 開発における技術課題

技術課題	備考(技術項目等)
運用技術	ITIL継続的改善

技術課題を ISO/IEC/IEEE29148([19])に掲載されている図 15 に位置付けた場合、要求の変化に関わる超上流工程（運用技術の一部も含む）の技術課題は、ビジネスオペレーションもシステムの状況もウォッチしていないと、システムオペレーションを作ることができない。報告書で活用している造語であるビジネス-IT オペレーションを担う層見出し、かつ具体的にシステムオペレーションにどう結び付ければ良いかについては検討課題としてとらえる。



ISO/IEC/IEEE29148[19]を参考に作成

図 15　ビジネスとシステムオペレーションの位置づけ

3.4. その他関連技術について

(1) SWEBOK2005 と対比による技術課題の全体像

図 16 のとおりである。超上流は現段階では SWEBOK では取り上げられていない。敢えて比較するとソフトウェア要求・設計、およびツールと方法としてとらえられる技術が多い（運用技術は技術課題がひとつのみのため除く）。非ウォーターフォール型開発技術は、自然とソフトウェア構築に置いたが、要求を固めるための技術としてとらえるとソフトウェア要求に置くべきかもしれない。プロセスで要求を固める技術と言える。

			◎: 技術課題すべてに○				○: 技術課題の一部に○				
技術課題（分類） SWEBoK知識項目 （一部システムに拡張）	超上流工程分析・評価	上流工程の横断的連携	要求管理	システム構成関連技術	安全と情報セキュリティ	運用技術	非ウォーターフォール型開発技術				
							要求の明確化	構成管理	短期開発	反復開発	品質管理
ソフトウェア要求			◎	○	○		○				
ソフトウェア設計				◎	○				○		
ソフトウェア構築				○	○		○	○	○	○	○
ソフトウェアテスト					○						
ソフトウェア維持					○	◎					
ソフトウェア構成管理								○			
ソフトウェア工学プロセス											
ソフトウェア工学ツールと方法		◎	○		○			○	○	○	
ソフトウェア品質				○							○
以下はSWEBoK範囲外											
超上流工程技術	◎					◎					

図 16　技術課題と SWEBOK の知識項目との関係

⑵ アーキテクチャフレームワークについて

アーキテクチャフレームワークは、重要な技術体系であり、それぞれで設定された観点は、アーキテクチャ分析軸として非常に重要な技術体系と言え、良質なアーキテクチャ構築に貢献するものと考える。良質なアーキテクチャを構築することは、技術課題ではあるが、一方で要求の変化への対応という観点では、一部を除いて特徴的であるとは言えない。一方で関連性があるものを含めることでは片手落ちと考えるため、本報告書では本章に取り上げることで技術課題に準ずる項目であると位置づける。ここでは次のアーキテクチャフレームワークについて簡単に説明する。

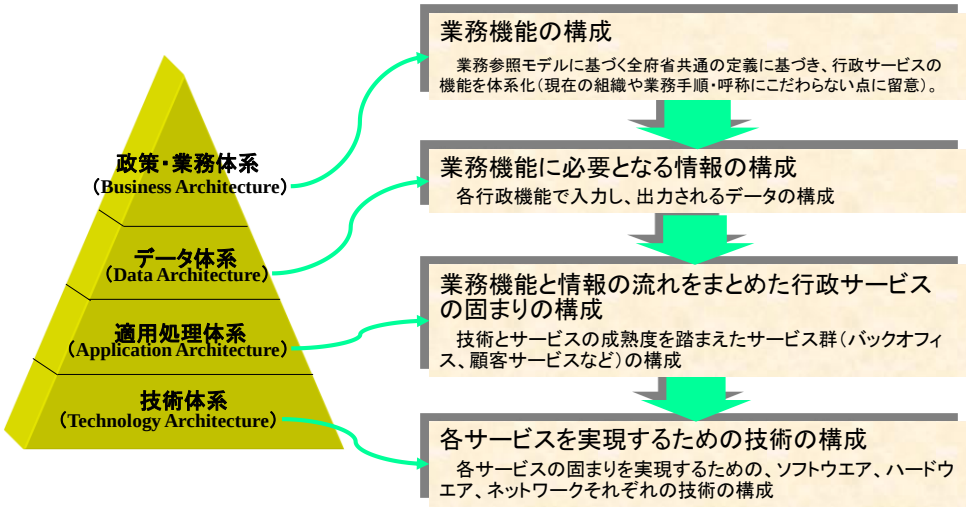
- 経済産業省の EA（Enterprise Architecture）
- Zachman アーキテクチャフレームワーク
- TOGAF
- DoDAF

なお、経済産業省の EA については、次項にてステークホルダー要求による整理の一環で EA の分類を活用する。また TOGAF については、ライフサイクルプロセスの一環で活用する。

① 経済産業省の EA

経済産業省では、自治体や省庁を中心に無駄のないシステム調達を実施するために EA というフレームワークを設定している。EA を構築する観点として図 17 のとおり、次の 4 つの観点を用意している。

- ビジネス参照モデル（BRM）
- データ参照モデル（DRM）
- サービスコンポーネント参照モデル（SCRM）
- 技術参照モデル（TRM）



出典： 経済産業省： Enterprise Architecture について、平成 16 年.

図 17 経済産業省の EA

② Zachman のアーキテクチャフレームワーク

アーキテクチャフレームワーク導入の初期段階で提案されたものである。歴史的な価値が高い。複数観点による分析をステークホルダーごとに行うことで分析するというアーキテクチャフレームワークの考え方を提唱したことは意義深い。その後他のアーキテクチャフレームワークのような実質的に活用されるものを生み出すきっかけになったことに意義がある。

表 2 Zachman アーキテクチャフレームワーク

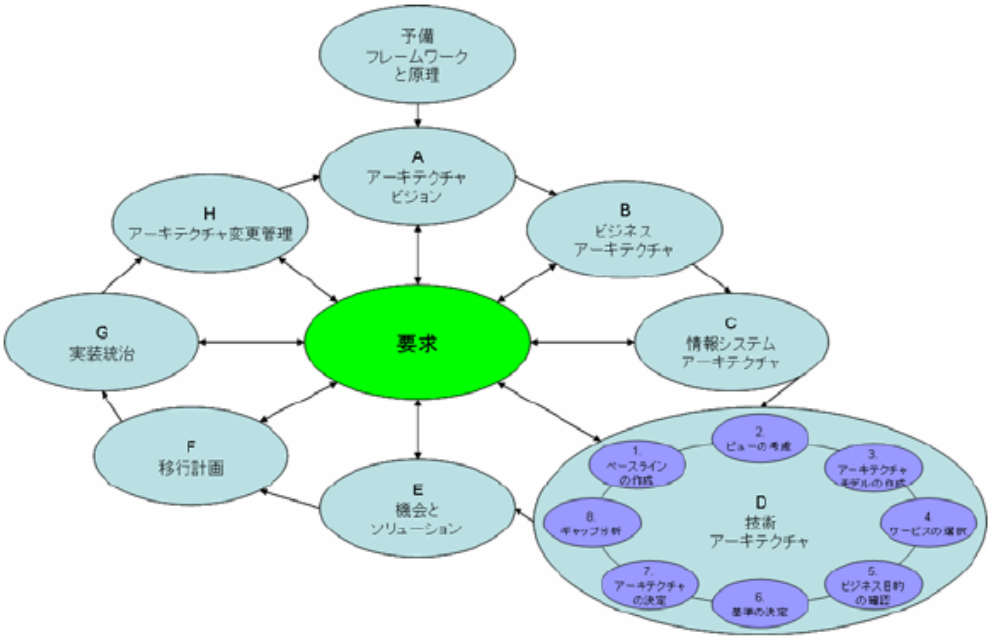
	what -data （データ）	how -function （機能）	where -network （場所）	who -peaple （組織・人）	when -time （時間）	why -motivation （動機）
Planner （計画立案者）	ビジネス重要項目の列挙	ビジネスプロセスの列挙	ビジネス物流拠点の列挙	ビジネス重要な組織の列挙	周期・行事・重要イベント列挙	ビジネスの目標と戦略
Owner （利用者）	意味的モデル	ビジネスプロセスモデル	拠点間物流システム	ワークフローモデル	マスタースケジュール	ビジネス計画
Designer （設計者）	論理データモデル	アプリケーション構造	分散システム構造	HMI構造	処理順序構造	ビジネスルールモデル
Builder （開発者）	物理データモデル	システム設計	テクノロジ構造	表現構造	制御構造	ルール設計
Subcontractor （作業者）	データ定義	プログラム	ネットワーク構造	セキュリティ構造	処理タイミング	ルール定義
FunctioningSystem （稼動システム）	データ	機能	ネットワーク	組織	スケジュール	戦略

出典： IPA: 参照アーキテクチャ報告書, 2005.

③ TOGAF Version9

TOGAF は、the Open Group により、とりまとめられているアーキテクチャフレームワークである。EA を含む形で非常に大きな体系を策定している。変容

への対応が特徴となっており、観点そのものがフェーズとしてとらえられ、フローにより変容への対応、改善ループが築かれている。TOGAF-ADM と呼ばれるアーキテクチャ開発手法では、図 18 のとおりのアーキテクチャ開発サイクルを描いている。各フェーズはプロセスとして本報告では扱い、次項のライフサイクルプロセスの分類で簡単な説明を行っている（図 28）。



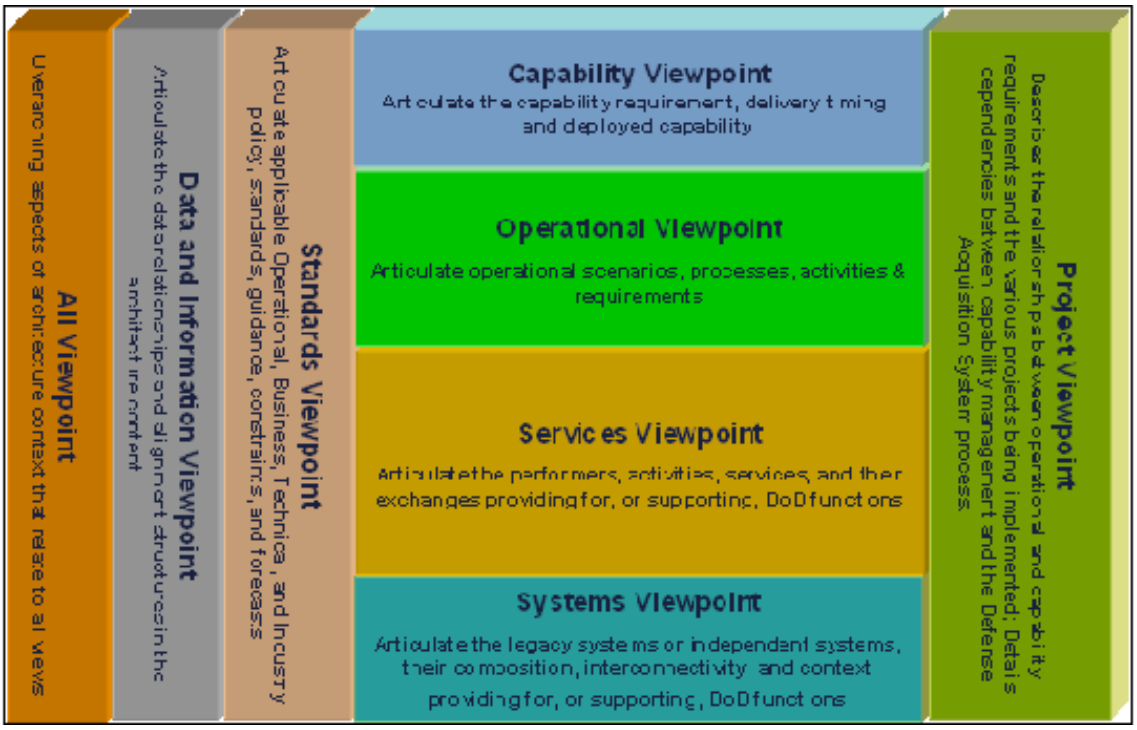
出典： グローバル情報社会研究所： TOGAF Version9 - 日本語訳 -

図 18 TOGAF-ADM のアーキテクチャ開発サイクル

④ DoDAF V2.0

DoDAF は DoD が定めたシステムズエンジニアリングのアーキテクチャフレームワークである。DoDAF V2.0 からは、データモデルを中心に説明されている。次の 8 つの観点を設定している（図 19 で説明されている）。

- All Viewpoint
- Capability Viewpoint
- Data and Information Viewpoint
- Operational Viewpoint
- Project Viewpoint
- Services Viewpoint
- Standards Viewpoint
- Systems Viewpoint



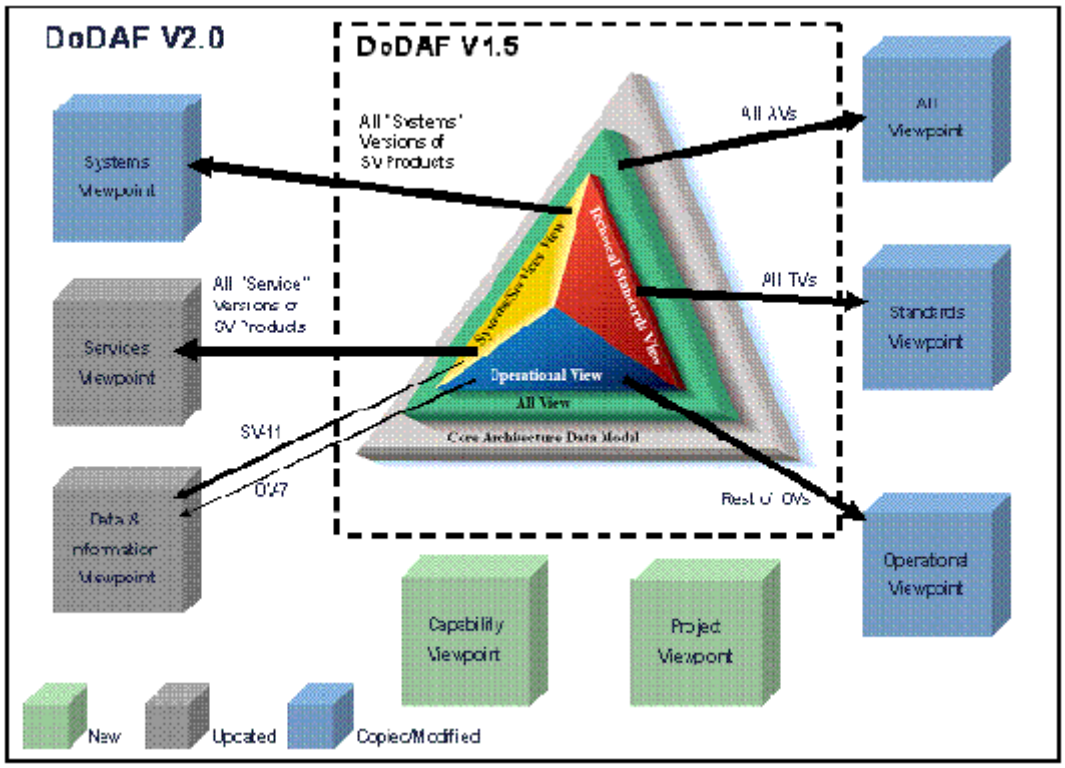
出典： 文献[11]

図 19 DoDAF2.01 の 8 つの観点

DoDAF は 1.5 版から 2.0 版になって観点が増えている（図 20）状況である。
参考までに DoDAF V1.5 の観点は次のとおりであった。

- All Viewpoint
- Operational Viewpoint
- Systems and Services Viewpoint
- Technical Standards Viewpoints

利用も進み、発展し続けていることがわかる。



出典： 文献[11]

図 20　DoDAF の V1.5 から V2.0 への発展

4. 技術課題項目の整理と深掘調査

4.1. 技術課題項目の整理の方針

技術課題を単純に並べることは、ピンポイントの項目が並んだだけでひとつのまとまったイメージにはならない。そのため技術課題を訴求することを前提とした整理軸を設定し、その上に技術課題をプロットして整理することを狙う。

整理軸は次のまとめ方をベースにした上で、改良を加えた。

① プロセスを中心とした整理と課題

- ライフサイクルプロセスによる整理（共通フレーム）

② 要求とアーキテクチャを中心とした整理と課題

- ステークホルダー要求を起点とした整理
- 品質特性と制約を起点とした整理
- アーキテクチャと構成要素を起点とした整理
- 山本 WG 主査の軸による整理

WG の議論を通じて、整理軸を再整理したまた IPA/SEC 内での報告会によるコメントを反映した。その概要は次のとおりである。

- **要求とアーキテクチャを中心とした整理**

  数が多いため整理軸としてまとめられるものはまとめる。山本 WG 主査の軸による整理はステークホルダー要求の整理の属性として表現した。

- **プロセスを中心とした整理**

  さらに次のように整理をした。

  - 開発工程（技術プロセス）

    システムが対象であるため、ISO/IEC15288:2008([16])を活用するため、一世代前の標準をベースとして共通フレーム 2007 ではなく、ISO/IEC12207:2008([15])を活用した。
  - 運用工程

    ITIL を活用することとなった。各段階をプロセスと見立てた。
  - 超上流工程

    要求の変化に限定した。エンタプライズシステムズエンジニアリングプロセスを活用した。
  - アーキテクチャと構成要素

具体的なシステムに整理の仕方が依存してしまう。技術の整理軸として分析の観点をもつアーキテクチャフレームワークを活用した。具体的にはTOGAFのフェーズをプロセスと見立てて整理軸とした。

・ **ステークホルダー要求による整理**

「ステークホルダー」の位置づけが様々な標準で異なっているため、再整理を行った。ただし要求による分類は例にしかならないことに留意した。技術項目としての整理というより、具体的なシステムの特徴付方法として位置づけた。

・ **品質特性による整理**

ステークホルダー要求による整理の一環としてまとめた。システムを対象とするため、ISO/IEC25010([18])を活用した。

結果的に整理軸は、次のとおりとなった。

プロセスを中心とした整理	
	超上流工程： Enterprise Systems Engineering Processes
	開発工程： ISO/IEC12207:2008, ISO/IEC15288:2008
	運用工程： ITIL(ISO/IEC20000)の各段階
	アーキテクチャフレームワーク： アーキテクチャ開発手法 TOGAF-ADM の各フェーズ
ステークホルダー要求を中心とした整理	
	ステークホルダー要求と技術項目
	品質特性 ISO/IEC25010 と制約による整理 レイヤ、山本 WG 主査の分類、経産省の EA の 参照モデルは属性として整理
その他の整理	
	品質要求と制約
	アーキテクチャと構成要素

4.2. ライフサイクルプロセスによる整理

超上流、開発および運用、それからアーキテクチャ開発手法として TOGAF-ADM のフェーズをプロセスとして扱って整理を行った。

(a) ISO/IEC12207:2008 および ISO/IEC15288:2008 のテクニカルプロセスによる分類

開発のライフサイクルプロセスによる分類を行う。システムが対象であるため、ISO/IEC15288([16])を活用しているが、ソフトウェア開発については、当初共通フレーム 2007 第 2 版が対象であったが、2008 年度版から ISO/IEC12207:2008([15]) に ISO/IEC15288:2008 を整合させたことから同一構造になった。そのためこの 2 つの標準のテクニカルプロセスに注目して整理軸を作成した。整理軸は表 3 のとおりである（プロセス名は JIS X160:2012（[21]）より）。

表 3　開発プロセス

ISO/IEC12207	ISO/IEC15288	プロセス
○	○	利害関係者要求定義
○	○	システム要求分析
○	○	システム方式設計
○	○	システム実装
○		ソフトウェア実装
○		ソフトウェア要求事項分析
○		ソフトウェア方式設計
○		ソフトウェア詳細設計
○		ソフトウェア構築
○		ソフトウェア結合
○		ソフトウェア適格性確認テスト
○		ソフトウェア導入
○		ソフトウェア受入れ支援
○		ソフトウェア運用
○		ソフトウェア保守
○		ソフトウェア廃棄
○	○	システム結合
	○	システム検証
	○	システム移行
	○	システム適格性確認
	○	システム運用
	○	システム保守
	○	システム廃棄

2 つの ISO の構造の差異部分については併記することとした。技術項目をプロットした結果は図 21 のとおりである。

【考察】

上流工程に多くの技術課題が固まってプロットされていることがわかる。

超上流工程の技術については、その結果を上流工程等で参照することになるためここでは○を付けている。向きが一方向であることに注意されたい。

また非ウォーターフォール型開発技術についてはソフトウェア構築技術のため、その部分にしていることもわかる。

全体に要求分析とアーキテクチャ設計を中心とした上流工程が重要であるがわかるが、このことは信頼性が高い情報システム構築と同様であった。

△ ひとつ以上が関係　○ 過半数が関係　◎ 全技術課題が関係

●:ISO/IEC12207+15288, ○ISO/IEC12207, ■: ISO/IEC15288

	プロセス	● 利害関係者要求定義	● システム要求分析	● システム方式設計	● システム実装	○ ソフトウェア実装	○ ソフトウェア要求事項分析	○ ソフトウェア方式設計	○ ソフトウェア詳細設計	○ ソフトウェア構築	○ ソフトウェア結合	○ ソフトウェア適格性確認テスト	○ ソフトウェア導入	○ ソフトウェア受入れ支援	○ ソフトウェア運用	○ ソフトウェア保守	○ ソフトウェア廃棄	● システム結合	■ システム検証	■ システム移行	■ システム適格性確認	■ システム運用	■ システム保守	■ システム廃棄
超上流工程分析・評価		◎	○	○	△	△	△	△	△	△	△	△						△	△	△	△			
上流工程における横断的連携		○	◎	◎	◎	◎	◎	◎	△	△	○	○						○	○	△	○			
要求管理		◎	◎	◎	◎	◎	◎	◎	△	△														
システム構成関連技術（部品化・連携）		△	◎	◎	◎	◎	◎	◎			△	△	△			△	△	○	○	○	○		△	△
安全と情報セキュリティ		◎	◎	◎	◎	◎	◎	◎	○	○	△	○			△	△	△	△	○	○	○	○	△	△
運用技術		○	○									○	○		○	○	○			○		○	○	○
非ウォーターフォール型開発技術	要求の明確化	○	○	△	○	◎	◎	○	△	○														
	構成管理					◎	△	△	△	◎														
	短期間での開発					◎	○	○	○	○	△													
	反復による開発					◎	△	△	△	○											△			
	品質管理					◎			△	◎														

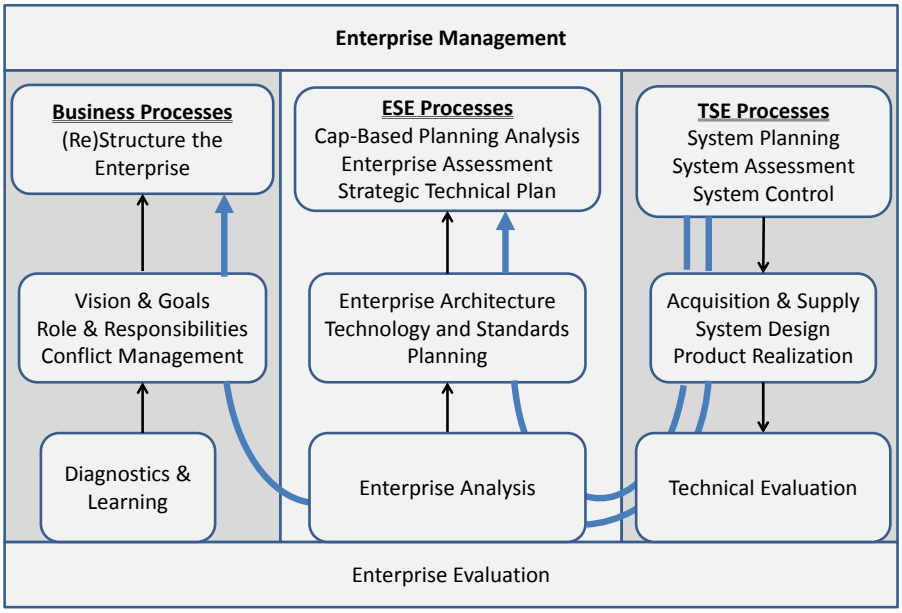
図 21　開発プロセスによる整理

(b) 超上流工程のプロセスによる分類

超上流工程として本調査ではシステムズエンジニアリングを調査対象としている関係で、エンタプライズシステムズエンジニアリングのプロセスを挙げている。そのため、そのままアクティビティにもなってしまう関係で同一項目となっていることをあらかじめ断っておく。

文献[1]および文献[4]によって提示されている主なプロセスは図 22 のとおりであり、全体では次のとおり記されている。当プロセスに重要な入力情報を提供する Stakeholder Analysis を追加した。

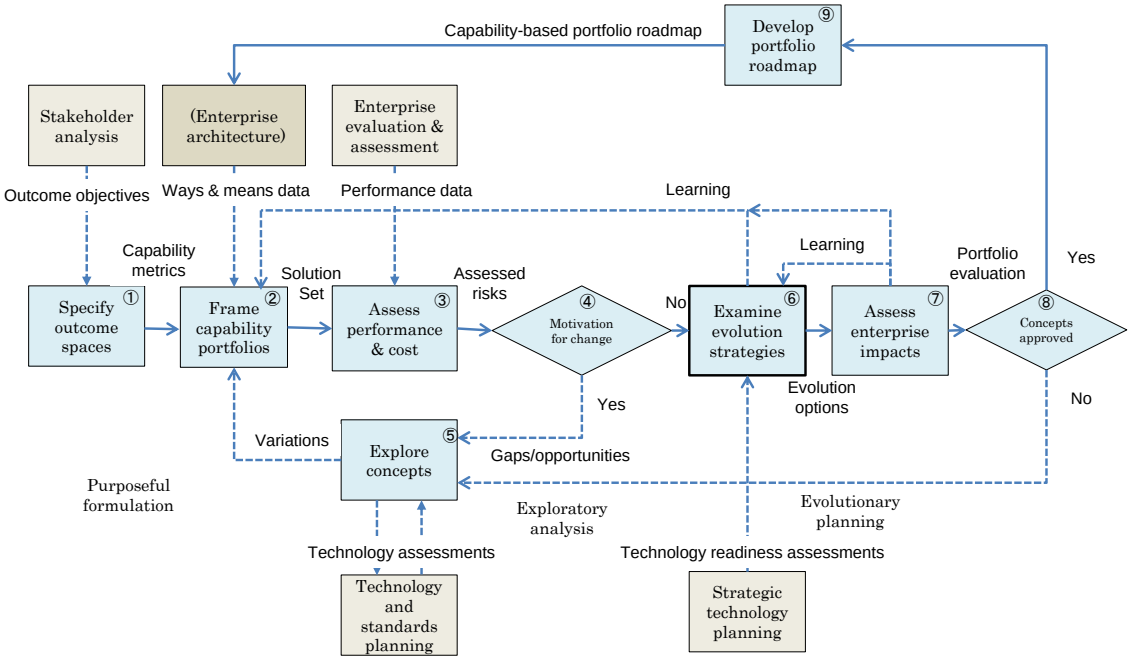
- 戦略的技術計画（Strategic Technical Planning）
- ケーパビリティに基づく開発の分析（Capability–based Planning Analysis）
- 技術と標準動向調査（Technology and Standards Planning）
- エンタプライズ有効性評価（Enterprise Evaluation and Assessment）
- ステークホルダー分析（Stakeholder Analysis）



出典： 文献[4]

図 22 ESE と既存開発との関係性例（再掲）

これらのプロセス間の関係例をに示す。



①～⑨までが、Capability-based planning analysis の機能である。

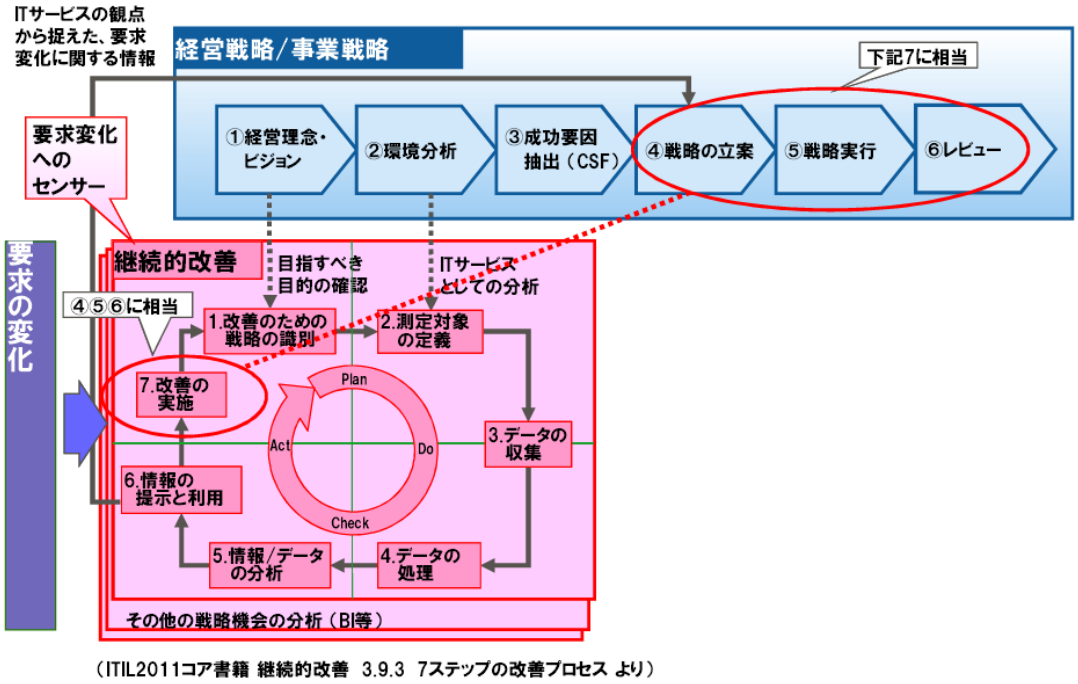
文献[1]より作成

図 23　ESE プロセスの関係性

技術項目をプロットしたものを図 26 に示す。

【考察】

図 26 のとおり、超上流工程のプロセスであるため、多くの技術課題が載らないが、超上流工程の技術項目がある意味独立していることを表しており、習得している技術者が少ないことが予想される。全体をコントロールする重要な責務もある工程であるため、**本項目の技術力強化が、IT 技術による産業力強化に貢献できることが想定できる。**一方で、運用技術の「継続的改善」としての技術課題が同プロセスでの解決も示しており、図 24 のとおり、WG での検討で挙げられているように経営戦略や事業戦略にも関係づけることができることから、**運用技術の継続的改善もまた同様な観点で重要であると考える。**

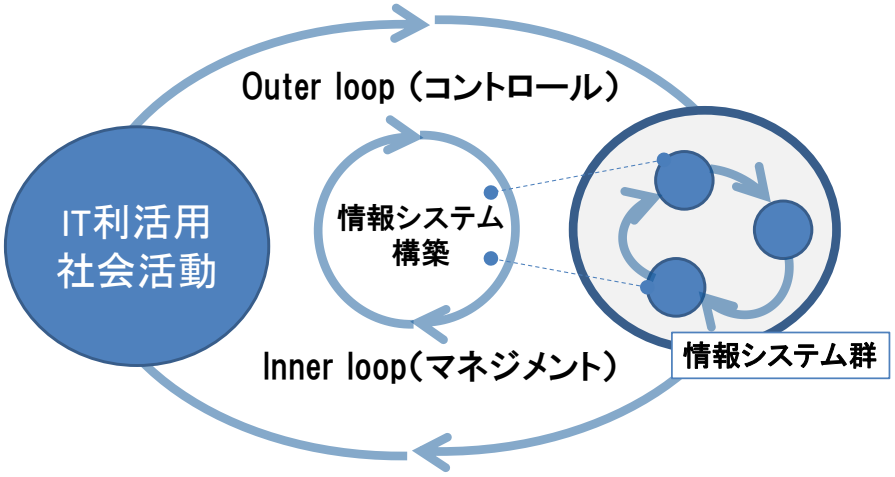


出典：WG 成瀬委員資料（itSMF Japan コンファレンス富士通講演より）([6])

図 24　IT サービスの観点での要求の変化への対応（図 14 の再掲）

併せて、**超上流工程では、3.1.1 や、文献[1]や図 25 に挙げているとおり、outer loop を形成していることが重要である。**その下で各システムの inner loop による対応が続くことになる。

これらの前提として、ケーパビリティを分析しておくことが求められる。今まであまり注目を浴びていなかったが、例えば実際組織ケーパビリティがないにも関わらず、過大な要求のシステムを構築してしまうと、実際には活用できない情報システムが構築され、いわゆる「動かないコンピュータ」の登場を招くことになる。どのようにケーパビリティの考え方を根付かせるかは検討課題と考える。



原出典：Hitchins, Derek, K.: Systems Engineering – A 21st Century Systems Methodology, John Wiley&Sons, 2007.

山本修一郎教授（WG 主査）ホームページ資料を基に作成

図 25　Outer Loop と Inner loop（システムズエンジニアリングの例）（図 8 の再掲）

超上流工程分析・評価のみ: ○ 関連プロセス ◎ 対応プロセス

△ ひとつ以上が関係 ○ 過半数が関係 ◎ 全技術課題が関係

	プロセス	戦略的技術計画 Strategic Technical Planning	ケーパビリティに基づく開発の分析 Capability-Based Planning Analysis	技術と標準動向調査 Technology and Starndards Planning	エンタプライズ有効性評価 Enterprise Evaluation and Assessment	ステークホルダ分析 Stakeholder analysis
超上流工程分析・評価	システムシンキング	○	○	○	○	○
	ケーパビリティに基づく開発の分析		◎			
	エンタプライズ有効性評価				◎	
	戦略的技術計画	◎				
	技術および標準化動向調査			◎		
	ステークホルダ分析					◎
	ConOpsの分析/定義	○	○	○	○	○
上流工程における横断的連携		△	△		○	△
要求管理						◎
システム構成関連技術（部品化・連携）				◎		
安全と情報セキュリティ				△		△
運用技術		◎	◎	◎	◎	◎
非ウォータフォール型開発技術	要求の明確化					
	構成管理					
	短期間での開発					
	反復による開発					
	品質管理					

図 26　超上流工程プロセスによる整理

(c) 運用プロセスによる整理

運用プロセスとしてはITILの各段階を、プロセスの代わりに本報告書では活用する。具体的な段階は**表 4**のとおりである。

表 4 ITIL V3 の各段階

山本WG主査ホームページ資料を基に作成

段階	説明	内容
サービス戦略	IT サービスマネジメント全体戦略を確立	財務管理、需要管理 戦略策定、 サービス・ポートフォリオ管理
サービス設計	事業要件に基づく品質、信頼性、柔軟性を満たすサービスを設計、サービス管理の仕組みと技術アーキテクチャ、プロセスを検討	サービスレベル管理、 キャパシティ管理、可用性管理、 IT サービス継続性管理、 サプライヤ管理 サービスカタログ管理、 情報セキュリティ管理
サービス移行	サービス設計に基づき事業ニーズの充足性をテストすることにより、本番環境に展開	変更管理、 サービス資産管理及び構成管理、 移行の計画立案とサポート、 リリース管理及び展開管理 ナレッジ管理、 サービスの妥当性確認及び テスト、評価
サービス運用	IT サービスを合意されたレベルで事業に提供	インシデント管理、問題管理、 サービスデスク イベント管理、要求実現、 アクセス管理、技術管理、 IT 運用管理、 アプリケーション管理
継続的サービス改善	IT サービスとIT サービスマネジメントを継続的に改善	サービス報告、サービス測定、 サービスレベル管理 7 段階の改善プロセス

当プロセスと要求の変化との対応は前述の図 24 のとおりであり、超上流工程技術としても位置付けられる（今回は運用技術のみ）。技術項目をプロットした結果は図 27 のとおりである。継続サービス改善にある意味集中が起こっている。このプロセスが要求の変化にとって重要なプロセスである。

【考察】

図 27 のとおり、超上流工程分析・評価に分類したエンタプライズシステムズエンジニアリングのプロセスから、上流工程の技術も、IT サービスの継続的改善プロセスに関連していることがわかる。

これは超上流工程の技術は、トータルな意味で情報システムのビジネス-IT オペレーションの一環としてをコントロールする役割を担っているため、運用技術側や、開発技術側からの課題を超上流工程技術への受け渡しをするという重大な役割をもっている。

△ ひとつ以上が関係　○ 過半数が関係　◎ 全技術課題が関係

ITIL V3 フェーズ		継続的サービス改善	サービス・ストラテジ	サービス・デザイン	サービス・トランジション	サービス・オペレーション
超上流工程分析・評価		◎	△	△	△	△
上流工程における横断的連携		△				△
要求管理		○				△
システム構成関連技術（部品化・連携）		△				
安全と情報セキュリティ		○	△			○
運用技術		◎	◎	◎	◎	◎
非ウォータフォール型開発技術	要求の明確化	△		△		△
	構成管理					
	短期間での開発	△		△		△
	反復による開発	△		△		△
	品質管理					

図 27　運用プロセスによる整理

(d) アーキテクチャ開発手法のプロセスによる整理

TOGAF-ADM（Architecture Development Method）のフェーズをプロセスとみなした整理である。本プロセスは WG の指摘により追加した。TOGAF-ADM は、変化を前提とした反復プロセスとみなすことできるが、AsIs から ToBe への変化であることに注意が必要である。

TOGAF-ADM のフェーズは次のとおりである。

- 準備: TOGAF アーキテクチャプロジェクトの準備活動
- A.アーキテクチャビジョン:
  スコープ、制約、期待、ステークホルダーを定義、事業環境を確認
- B.ビジネスアーキテクチャ:
  ビジネスの現行と目標アーキテクチャを定義、差異を分析
- C.情報システムアーキテクチャ:
  情報システムの現行と目標アーキテクチャを定義、差異を分析
- D.技術アーキテクチャ:
  技術の現行と目標アーキテクチャを定義、差異を分析
- E.ソリューション:
  実施計画、展開手段、要素を定義し移行アーキテクチャを構築
- F.移行計画:
  費用対効果分析、リスク分析に基づき移行実施計画を詳細化
- G.実装監督:
  アーキテクチャ移行計画を管理、実装結果を確認
- H.アーキテクチャ変更管理:
  アーキテクチャの事業目標適合性を継続的監視、変更管理
- 要求管理: 全工程で要求確認を実施

簡単にプロットした結果を図 28 に示す。

【考察】

超上流工程プロセスによる整理と似ているところがある上、上流工程の技術も含んでいることがわかる。また運用プロセスについても、個別システムという意味で十分に関係づけることができる。

ただし、超上流工程の技術課題については、現在のところ関係づけられる可能性に言及するに留める。アーキテクチャ開発手法は、それで詳細な方法論となっているため、他の技術課題との混在可能性については検討の余地がある。変化に対応するための技術体系として位置づけておくことに留める。

△ ひとつ以上が関係　○ 過半数が関係　◎ 全技術課題が関係

TOGAF-ADM		初期フェーズ	アーキテクチャビジョン	ビジネスアーキテクチャ	情報システムアーキテクチャ	技術アーキテクチャ	ソリューション	移行計画	実装監督	アーキテクチャ変更管理	要求管理
超上流工程分析・評価		○	◎	◎	○	○	○	○	○	○	△
上流工程における横断的連携			△	○	◎	△	○			○	○
要求管理			△	△	△	△	△			△	○
システム構成関連技術（部品化・連携）		△	△	△	◎	◎	◎	△	△	△	
安全と情報セキュリティ		△	○	○	△	△	○	△	△	○	△
運用技術				◎	◎	◎	◎				◎
非ウォータフォール型開発技術	要求の明確化									○	○
	構成管理										
	短期間での開発										
	反復による開発										
	品質管理										

図 28　アーキテクチャ開発のプロセスによる整理

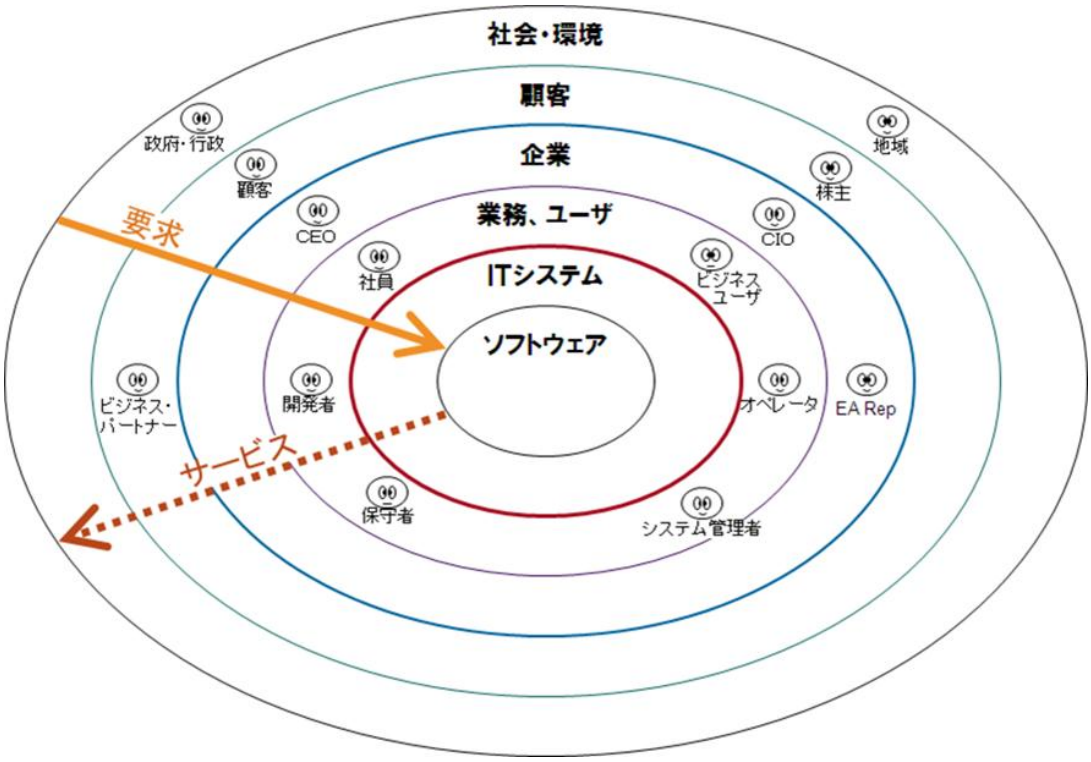
4.3. ステークホルダー要求による整理

ステークホルダー要求による整理を行った。以下、ステークホルダーの分析結果を示した後、技術課題とステークホルダーとの関係例、さらに技術課題と品質特性との関係例を示す。なお、品質特性の整理においては他の当初予定した軸も次のように含めることとした。

- 品質特性はシステムおよびソフトウェア向けに ISO/IEC25010([18])を活用した。
- 山本主査の分類は、各技術課題の属性として表現した。
- レイヤについても、各技術課題の属性として表現した。
- アーキテクチャは、経産省の EA を属性として表現した（なお EA などのアーキテクチャフレームワークについては付録にて解説）。

(a) ステークホルダーについての分析

ステークホルダーをうまく表現した例として KIKUMA RING を図 29 に示す。



IT アーキテクチャサミット 2008，日本 IBM 榊原氏資料を一部改編

図 29　KIKUMA RING

円の各層に様々なステークホルダーを表現できる。この図の中だけでも多様なステークホルダーがいる。ステークホルダー要求といっても注意が必要であることがわかる。ステークホルダー要求を WG において各種資料で分析した結果が表 5 のとおりである。

表 6 に、WG によって検討された、各種ユーザ企業を中心に実施されたアンケート調査の IT 投資および保守理由に関する設問に回答より抜き出した結果に対して、表 5 から導出したステークホルダーとの関係性を分析し、要求の変化に関係するステークホルダー要求を導出した結果を掲載する。

【考察】

「ステークホルダー要求」が情報システム変化の主要原因として、各種の標準で認識されているにもかかわらず、標準間で必ずしも統一的な定義がなされていないため、整理が必要であると判断した。このような、各種の標準におけるステークホルダー要求の横断的な比較結果は、本調査が世界的にも最初である可能性があり、この結果だけでも非常に価値がある。

具体的には、表 5 よりステークホルダーの認識による違いがわかる。要求の変化への対応という面では運営技術に着目すると、共通フレーム、REBOK が開発技術に重点が置かれていると解釈できる。要求の変化に対応するイメージは継続した開発、文献[1]によるとシステムのデザイナが継続して存在することが求められており、また WG での議論では要求管理者が継続的に存在することが前提となる。PMBOK にはポートフォリオマネージャやプログラムマネージャ等が存在するが、今回洗い出した超上流工程における要求変化との関係性について、さらに検討が必要となる。特に要求管理者としてのステークホルダーとしての位置を定めることは課題である。[11]

他方、表 6 にあるように、各種調査の結果から経営目標に準ずるステークホルダー要求がそれなりに導出できることから、超上流工程を重視したシステムの実現は、ユーザ企業の方向性に適合していると考える。

ITIL 適用活用目的に関しては、2009 年当時の調査であることから、ITIL V2 を念頭においた調査であったことが推察されるため、運用面に関する要求が導出されていると考える。

現在、様々な問題抱えている局面では、その問題を解決するために複数の知識体系（標準類）を組み合わせて活用するケースがある。そのためには、表 5 に見られるようなステークホルダーに関する統一に向けた知識が必要となる。ステークホルダー要求からの洗い出しが増えている現在、このような表をベースとした体系化が重要となる。

---

11 組込みスキル標準（ETSS)では、プロダクトライン等による開発が前提となるため、PMBOK 以外の知識に基づくスキルも求められている。

表 5 共通フレームの分類をベースに各知識体系、フレームワーク間の関係を比較

ｽﾃｰｸﾎﾙﾀﾞｰの大分類	共通フレーム		BABOK		REBOK		ITIL	TOGAF		PMBOK	KIKUMA RING
	部署等／役割(ロール)		ステークホルダー		ステークホルダー		利害関係者	ステークホルダー		ステークホルダー	ｽﾃｰｸﾎﾙﾀﾞ
	分類	対象	分類	対象	分類	対象	対象	分類	対象	対象	対象
外部			影響を受ける 外部	顧客 規制者 サプライヤ			事業顧客/外部顧客 マスコミ 政府/規制機関 労働組合 株主 事業パートナ	外部	規制機関 調達先	顧客	政府行政 地域 顧客 ビジネスパートナ 株主
経営層	経営層	社長 担当役員	組織や企業	スポンサー 経営幹部	ユーザ	経営者(CIO)	トップ・マネジメント	本社機能	CxO	スポンサー	CEO,CIO, EA Rep
管理部門							監査人	本社機能	ｴﾝﾀﾌﾟﾗｲｽﾞｾｷｭﾘﾃｨ ﾌﾟﾛｸﾞﾗﾑﾏﾈｼﾞﾒﾝﾄ 品質保証/標準ｸﾞﾙｰﾌﾟ 調達／人事	影響力者	
業務部門	業務部門	部門長 業務推進担当 ｼｽﾃﾑ推進担当 関連会社	組織や企業 影響を受ける 組織ユニット	ドメインのSME ﾋﾞｼﾞﾈｽｱﾅﾘｽﾄ エンドユーザー	ユーザ	エンドユーザ	内部顧客 ユーザ 事業領域/事業部門	ｴﾝﾄﾞﾕｰｻﾞ 組織	役員 部門管理者 ﾋﾞｼﾞﾈｽ・ﾄﾞﾒｲﾝ専門家 データ所有者	ユーザー	社員 ﾋﾞｼﾞﾈｽﾕｰｻﾞ
情報システム部門	情報システム部門	部門長 ｼｽﾃﾑ開発担当 ｼｽﾃﾑ子会社	ｿﾘｭｰｼｮﾝの デリバリ	ﾌﾟﾛｼﾞｪｸﾄﾏﾈｼﾞｬｰ 実装のSME テスト担当者	ユーザ	ﾌﾟﾛｼﾞｪｸﾄﾏﾈｼﾞｬ システム責任者	内部サービスプロバイダ プログラムとプロジェクトチーム	ﾌﾟﾛｼﾞｪｸﾄ 組織	役員 部門管理者 ﾋﾞｼﾞﾈｽ・ﾌﾟﾛｾｽ専門家 ﾌﾟﾛｾｽ生成物専門家 技術専門家	ﾌﾟﾛｼﾞｪｸﾄ・ﾏﾈｰｼﾞｬｰ 母体組織 ﾌﾟﾛｼﾞｪｸﾄ・ﾁｰﾑ・ﾒﾝﾊﾞｰ ﾌﾟﾛｼﾞｪｸﾄﾏﾈｼﾞﾒﾝﾄ・ﾁｰﾑ PMO	開発者 保守者 ｼｽﾃﾑ管理者
ｼｽﾃﾑ運用部門			影響を受ける 組織ユニット	運用サポート ヘルプデスク			運用スタッフ	システム運用	ITサービス管理者 サービス・デスク ｱﾌﾟﾘｹｰｼｮﾝ管理者 ｲﾝﾌﾗｽﾄﾗｸﾁｬ管理者 ﾃﾞｰﾀ/音声ｺﾐｭﾆｹｰｼｮﾝ		ｵﾍﾟﾚｰﾀ
ベンダ	ベンダ	元請けベンダ アウトソーサ サブベンダ			ベンダ	ﾍﾞﾝﾀﾞ側経営者 ﾌﾟﾛｼﾞｪｸﾄﾏﾈｼﾞｬ 上級ｼｽﾃﾑｴﾝｼﾞﾆｱ ｿﾌﾄｳｪｱ開発者	サプライヤ(製品) サプライヤ(サービス) サードパーティ				

注）ITIL については、ITIL V3 (2007 年版)のコア書籍 ST の p.174 図 5.5 および図 5.6 からの引用である。

表 6　ステークホルダー要求例の分析結果

要求目的例 （IT投資効果）	①JUAS企業IT動向調査2011	②ｿﾌﾄｳｪｱﾒﾄﾘｯｸｽ調査2011	②ｿﾌﾄｳｪｱﾒﾄﾘｯｸｽ調査2009	ステークホルダー大分類					
	Q3-1.IT 投資で解決したい中期的な経営課題	Q3.2.保守作業の発生目的	Q6.2 ITIL適用活用目的	外部	経営層	管 理 部門	業 務 部門	情シス 部門	運 用 部門
企業の社会的責任 （CSR向上）	13.企業としての社会的責任の履行 （ｾｷｭﾘﾃｨ確保、個人情報の保護等）			○	○	○			
法令順守・業界標準への準拠		制度ルール変化	4.遵法あるいは認証取得	○	○		○	○	
経営の透明性確保	12.経営の透明性の確保 （内部統制、ｼｽﾃﾑ監査への対応等）		3.IT統制,ITｶﾞﾊﾞﾅﾝｽの強化		○	○			
経営スピード向上	1.迅速な業績把握、情報把握 （リアルタイム経営）	経営目標変化	1.経営あるいは事業からの要求		○	○	○		
市場シェア拡大	3.グローバル化への対応 10.ビジネスモデルの変革		2.ｸﾞﾛｰﾊﾞﾙ標準の採用		○		○		
顧客の確保、維持	2.顧客重視の経営 11.営業力の強化				○		○		
業務効率向上 （省力化、業務コスト削減）	4.社内コミュニケーションの強化 6.IT 開発・運用のコスト削減 7.業務プロセスの効率化 （省力化、業務コスト削減）	業務方法変化 ユーザビリティ変化 担当者要望	8.ITｻｰﾋﾞｽ管理の効率化 9.ITｻｰﾋﾞｽｺｽﾄの削減		○	○	○	○	○
業務スピード向上 （リードタイム短縮等）	5.企業間（グループ、業界、取引先間）の情報連携 8.業務プロセスのスピードアップ （リードタイム短縮等）		6.部門間連携の強化		○		○		
業務品質・精度の向上	9.業務プロセスの質・精度の向上 （ミス、欠品削減等）	システムバグ	5.ﾌﾟﾛｾｽ、ルールの明確化 7.ITｻｰﾋﾞｽの品質・信頼性向上				○	○	○
製品・ｻｰﾋﾞｽの開発、改善			10.IT運用に関する特定の問題解決						○
ITインフラ変更への対応 （製品保守、ﾃﾞｰﾀ量対応、最新技術の活用）		データ量の変化 ﾊｰﾄﾞ・ﾐﾄﾞﾙ変更への対応 OS変更への対応						○	○

(b) ステークホルダー要求と技術課題の例

前項で示したステークホルダー要求に対して、技術課題と対応付した例を表 7 に示す。

表 7　ステークホルダー要求と技術課題の対応例

要求（変化の要因）	ステークホルダー	技術課題例	
企業の社会的責任（CSR 向上）	システム部門	安全と情報セキュリティ	共通基盤としての保証
法令順守・業界標準への準拠	システム部門	安全と情報セキュリティ	他者に求められる保証
経営の透明性確保	経営企画部門	ESE	ケーパビリティに基づく開発に関する分析
経営スピード向上 顧客の確保、維持	システム部門	上流工程	機能に基づく SE 手法活用
市場シェア拡大	システム部門	システム構成技術	構造の明確化
業務効率向上 （省力化、業務コスト削減）	システム部門	運用技術	ITIL 継続的改善
業務品質・精度の向上	経営企画部門	ESE	ステークホルダー分析
製品・サービスの開発、改善	経営企画部門	ESE	ConOps の分析/定義
IT インフラ変更への対応 （製品保守、データ量対応、最新技術の活用）	システム部門	ESE	エンタプライズ有効性評価

この結果を基に、品質特性等による特徴付例として整理した結果を図 30 に示す。本整理結果について以下のとおり補足する。

- 例示のため、詳細な特徴を示してはいない。
- 大雑把な特徴が見えるが、本整理はそれよりも具体的なシステムの要求を適用するためのフレームワークとしては非常に有効であり、システムの特徴付をすることができる。

項番	ステークホルダ要求			技術課題	レイヤ		山本分類				要求分類					EA	ISO25010システム/ソフトウェア品質特性												制約
	要求	ステークホルダ	補足説明		現実世界	情報システム世界	適応	修正	変革	改良	アプリケーション機能	技術機能	ハードウェア機能	アーキテクチャ要求	IT外要求	③①ビジネス・②データ・適用処理・④技術	機能適合性	性能効率性	互換性	使用性	信頼性	セキュリティ	保守性	移植性	有効性	効率性	満足性	リスク回避性	コンテクストカバレッジ	
①	企業の社会的責任(CSR向上)	経営企画部門	技術戦略を検討してシステムへの適用を計画する。	ESE技術活用戦略 (Strategic Technology Plannning)	○	○										④													
②	法令順守・業界標準への準拠	システム部門	情報セキュリティが確保できることを保証するために、認証などの他者に求められる保証が必要となることがある。	品質保証:他者に求められる保証	○	○	○				○	○	○	○	○	④						○							
②	法令順守・業界標準への準拠	サービス提供企業	環境の変化から要求は絶えず進化するため、不測の事態を考慮し、ソフトウェアを進化させる必要がある。	要求管理:不測事態への対応、仮定の管理	○		○							○	○	①					○							○	
③	経営の透明性確保	経営企画部門	ステークホルダ要求の変化に対応した開発を許すことができるシステムを導入するために、そのシステムに必要とする能力を事前に把握するために全体プログラムの分析を行う。	ESE ケーパビリティに基づく開発に関する分析	○	○										①													
④	経営スピード向上 顧客の確保、維持	システム部門	結果として振舞、コンテキスト、コントロールフロー、データフロー、データディクショナリ、ER、Functional Flow Block、Models、Simulation Results、Integrated Definition for Functional Modelingの各図を得る。	上流工程 機能に基づくSE手法活用 Functions-Based Systems Engineering	○	○	○	○	○	○	○		○	○		③	○	○					○						○	
⑤	市場シェア拡大	システム部門	情報システムの保守性を良くするためには、ソフトウェアを分解可能な形で部品化しておく必要があり、そのためにはアーキテクチャ記述言語を活用して、構造の明確化と定義を行い、アーキテクチャを明確にする必要がある。	システム構成:構造の明確化		○	○	○	○	○	○			○		④							○	○						
⑤	市場シェア拡大	サービス提供企業	保守性を高めるために、拡張ポイントを明確にするためにスコープ決めを行う必要がある。	要求管理:スコープ決め		○	○				○					①	○	○											
⑥	経営スピード向上 顧客の確保、維持	システム部門	システムズエンジニアリングのプロセスをOMGが中心となってSysMLで記述する方法論を用意している。リアルタイム系の場合 Marteも活用することがある。なお、アシュアランスについても記述する方法論の研究も進んでいる。アクティビティとしては次のとおり。 Analyze Needs, Define Systems Requirements, Define Logical Architecture, Synthesize Allocated Architectures, Optimize and Evaluation Alternatives, Validate and Verify System	上流工程 オブジェクト指向SE手法活用 Object-Oriented Systems Engineering Method	○	○	○	○	○	○	○		○	○		③	○	○					○						○	
⑥	経営スピード向上 顧客の確保、維持	サービス提供企業	情報システムを保守し、要求を明確化し、見えるような状態にしておき、変化に耐えるようにする必要がある。	要求管理:要求の明確化	○	○	○	○	○	○																			
⑥	経営スピード向上 顧客の確保、維持	サービス主管部門	顧客のニーズに従った商品を適時に提供するため、システムを絶えず進化させるための前提として要求の変化を明確化する必要がある。そのため、要求を明確化する技術を必要とする。	要求の明確化	○	○	○	○	○	○	○			○		③	○													
⑦	業務効率向上(省力化、業務コスト削減)	システム部門	環境の違い等の少々の変化による改修を避けるために、冗長性を確保したソフトウェアの実現を確認するロバストネステストを実施する。	テスト技術:ロバストネスの確保		○	○		○		○			○		③	○		○				○	○						
⑧	業務スピード向上 (リードタイム短縮等)	経営企画部門	システム全体のもつべき全体の統合の在り方を分析し、そのコンポーネントとアーキテクチャ構成を診断する。	ESE システムのあるべき全体統合の調査・分析(Enterprise Analysis and Assesment)	○	○										①													
⑨	業務品質・精度の向上	経営企画部門	当該システムのステークホルダが誰であり、どのような役割をもっているのかを分析する。	ESEステークホルダ分析 (Stakeholder analysis)	○	○										①													
⑨	業務品質・精度の向上	システム部門	システムズエンジニアリングに限らず、ソフトウェアエンジニアリングでも重要な技術である。変化した要求への対応について上流工程段階で設計検証することは非常に重要である。基本的な手法として位置づける。	上流工程 モデリング、シミュレーション、プロトタイピング	○	○	○	○	○	○	○	○	○	○		③	○	○					○							
⑩	製品・ｻｰﾋﾞｽの開発、改善	経営企画部門	システム全体としての Operation の在り方と、それに従ったシステムのコンポーネントのOperationの在り方、相互依存関係などを分析して、システムのOperation の概念をまとめる。	ESE CONOPSの分析/定義	○											①													
⑪	ITインフラ変更への対応(製品保守、ﾃﾞｰﾀ量対応、最新技術の活用)	経営企画部門	今後の技術動向を予測する。	ESE技術および標準動向調査 (Technology planning)		○										④													
⑪	ITインフラ変更への対応(製品保守、ﾃﾞｰﾀ量対応、最新技術の活用)	システム部門	データ量の変化や、ハード・ミドル変更の対応、OS変更の対応など将来変わる可能性がある。	コントロールケースによる非機能要求記述	○	○	○		○					○		①					○		○	○			○		

図 30　ステークホルダー要求を通した技術課題と品質特性の関係例

(2-3) その他の整理

技術課題の品質特性と制約の観点による整理、技術課題のアーキテクチャと構成要素の観点による整理結果を図 31 および図 32 に示す。いずれも例によるアーキテクチャと構成要素による分類は、文献[12]による整理を活用した。

【分析】

両者ともにあくまで事例ということになる。

前者の品質特性による技術課題の整理は、要求分類としてアプリケーション要求が多くなっていることから、機能適合性が多く求められている。同時に保守性とリスク回避性も求められている。主に超上流、上流工程に全体がシフトしていることに起因する。超上流工程の技術課題は当然なことである全品質特性的に影響がある可能性を秘めており、このあたりはケーパビリティをどのように定義するのかにも左右される場合もある。

アーキテクチャと構成要素による技術課題の整理については、文献[12]にて Patterns for e-business（[14]） のセルフサービス型のアーキテクチャパターンに基づいて分類した結果に基づいている。このレベルの技術課題については、どの基本パターンにも関係付けられるが、基本的なアーキテクチャとしてプレゼンテーション層とバックエンドのアプリケーション層の分離が基本となっている。アーキテクチャパターンとして、スタンドアロンシングルチャネル、ルータ、エージェントなどが挙げられている。要求の変化として M&A 絡みの要求を例示しているところから、アーキテクチャパターンとしてルータなどの選択が中心となるため、直接結合と関係の基本パターンとの関係性が低くなっている。

このような基本パターンを収集することで、要求の変化に向いたアーキテクチャパターンが見つけやすくなる可能性がある。

△ ひとつ以上が関係　○ 過半数が関係　◎ 全技術課題が関係

	ステークホルダ要求		要求分類					EA	ISO25010システム/ソフトウェア品質特性														
技術課題	要求例	ステークホルダ	アプリケーション機能	技術機能	ハードウェア機能	アーキテクチャ要求	IT外要求	①ビジネス・②データ・③適用処理・④技術	機能適合性	性能効率性	互換性	使用性	信頼性	セキュリティ	保守性	移植性	有効性	効率性	満足性	リスク回避性	コンテクストカバレッジ	制約
超上流工程分析・評価	モデリングを活用した迅速な開発 企業の社会的責任（CSR向上） 業務品質・精度の向上 製品・ｻｰﾋﾞｽの開発、改善 ITインフラ変更への対応（製品保守、ﾃﾞｰﾀ量対応、最新技術の活用）	経営企画部門 システム部門						① ④	○	○	△	○	○	○	○	△	○	△	○	◎		○
上流工程における横断的連携	経営スピード向上 顧客の確保、維持 迅速な開発、顧客確保、維持 業務効率向上（省力化、業務コスト削減） 業務品質・精度の向上	システム部門	◎	△	○	○		① ③	◎	○	△				◎	△						
要求管理	市場シェア拡大 経営スピード向上 顧客の確保、維持 ITインフラ変更への対応（製品保守、ﾃﾞｰﾀ量対応、最新技術の活用）	サービス提供企業 サービス主管部門 システム部門	○	△	△	○	△	① ③	○	△			△	△	○	△				△		
システム構成関連技術	市場シェア拡大 経営スピード向上	システム部門	◎			◎		③ ④	○	△	△		○		◎	△					△	○
安全と情報セキュリティ	企業の社会的責任（CSR向上） 法令順守・業界標準への準拠	サービス提供企業 システム部門	○	△	△	○	○	① ③ ④	△				○	○	△				△	△		△
運用技術	経営スピード向上	システム部門	◎				◎	①									◎	◎	◎	◎		
非ウォーターフォール型開発技術	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	◎					④	◎	△		△	△		△				○			

図 31　技術項目の品質特性による整理

△ ひとつ以上が関係　○ 過半数が関係　◎ 全技術課題が関係

技術課題	ステークホルダ要求	静的構造				動的振舞				要求分類				
	要求例	アプリ層とプレゼン層の分離	直接接続	ハブ・アンド・スポーク接続	更新データのハブ配置	プレゼン層の同期統合	プレゼン層の非同期統合	アプリ層の同期統合	アプリ層の非同期統合	アプリケーション機能	技術機能	ハードウェア機能	アーキテクチャ要求	IT外要求
超上流工程	e-business向けシステムの効率的構築 TCOの最小化と保守性の向上 バックエンドテンアプリケーションの統合化 迅速なビジネス開始 複数のデリバリー・チャネルの統合 顧客から見たビューの統一	◎	△	○	△	○	○	○	△					
上流工程連携	バックエンドテンアプリケーションの統合化 迅速なビジネス開始 ツール活用による開発コスト削減と早期開発	◎	△	◎	◎	○	◎	○	◎	◎	△	○	○	△
要求管理	バックエンドテンアプリケーションの統合化 保守性の向上 一括カスタマイズ	◎	△	○	○	○	○	○	○	○	○	○	◎	○
システム構成技術	保守性の向上 ITインフラ変更への対応（製品保守、データ量対応、最新技術の活用）	◎		○	○	○	○	○	○	◎			◎	
安全と情報セキュリティ	法令順守・業界標準への準拠 企業の社会的責任（CSR向上）	◎		◎	◎	◎	○	○	○	○	△	△	○	○
運用技術	経営スピード向上	◎		◎	◎	◎	◎	◎	◎	◎				◎
非ウォーターフォール型開発技術	業務品質・精度の向上 製品・サービスの開発、改善	○	△	○	○	○	○	△	△	◎				

図 32　技術課題のアーキテクチャと構成要素による整理

4.4. 技術課題の分析

前項までの結果を分析する。分析の方向性としてIPA/SECとしての技術課題の解決方法および普及展開方法の検討の一環として次の点に注目することとした。

- 超上流工程技術の展開に向けた分析

前項の結果を踏まえて、技術傾向を表現する方法としてプロセスによる整理を活用する。

また、情報システムの継続性についての技術を体系化する上で、これまでの取り組みが個別的であったことから、技術の整理軸が断片的で，相互の関係が不明確であった。この結果、継続性技術について、これまで十分な整理が進んでいなかった。複数の整理軸を活用した分析も検討した。

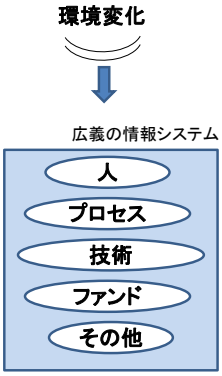
(1) 要求の変化を前提とするシステム構築技術像の分析

1.1(5)のとおり、要求の変化を前提とするシステム構築技術は、よりコントロール指向のシステム構築としてとらえられる。超上流工程技術や運用技術を活用してまとめる。

コントロール思考を担う超上流工程は、単体システム（もしくはSystem of Systems）の上流工程の前工程という位置づけではなく、環境の変化の影響を受ける事業全体に関連したシステムの集合体を対象としている。

超上流工程の技術課題はビジネスオペレーション中のITに関わる部分（本報告書ではビジネス-ITオペレーションと呼んでいる）に位置付けられる。エンタプライズエンタプライズシステムズエンジニアリング的な視点で考えると、変化の影響を被る対象は「広義の情報システム」であり、人（組織）、プロセス、技術からファンドまでも含まれることに留意したい（図 33）。

単体システムの要求を変化させなくてはいけない原因を探って、その影響を受けるということではなく、超上流工程から環境の変化の影響を考え、それを単体システムに反映していくという考え方であり、そのためには、コントロール指向のシステム構築技術が必要となる。



文献[1]の Fig2-2 を参考に作成

図 33　環境の変化の影響を被る対象（図 2 の再掲）

要求が変化したことを従来型の保守開発の一環として対処している場合の多くは、ボトムアップ指向でありマネジメントにより解決している。他方、要求の変化を前提とするシステムの開発はトップダウン指向でコントロールにより解決すると解釈できる。

このようなトップダウン指向で考える場合、重要なのは図 34 のとおり、超上流工程はひとつの改善サイクルが回る必要があり（Outer loop のことである）、個々のシステム構築工程も要求の変化に対応すべく改善サイクルが回る必要がある（Inner loop のことである）という 2 重構造を描くことになる。

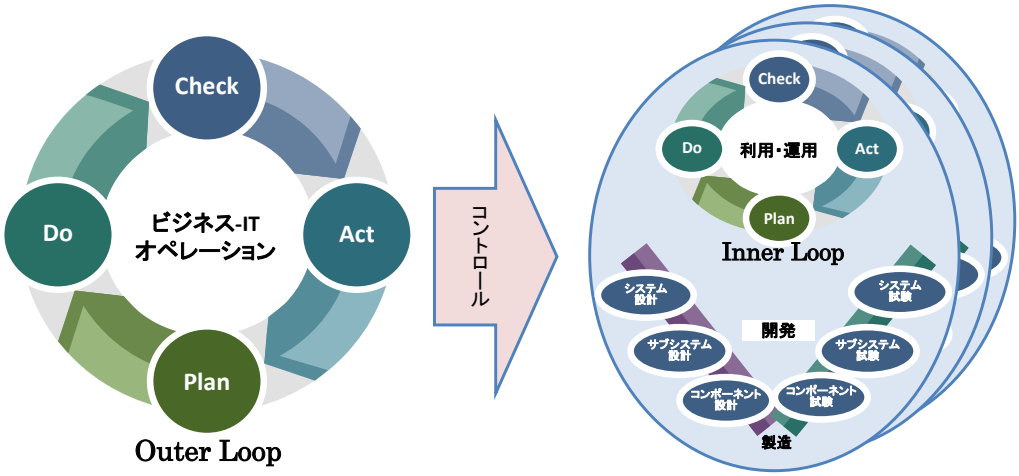


図 34　Outer Loop と Inner loop（システムズエンジニアリングの例）

注意すべきこととして、Outer Loop は単体システムではなく複数のシステム（人等も含む）対象であり、Inner Loop は複数存在することである。Outer Loop の改善は主にケーパビリティの見直しとなる。Inner Loop からのモニタリング情報やビジネスモニタリング情報を分析し、現行のケーパビリティに課題がある場合は、改善する必要が

あり、情報システム全体に影響する（図 35）。一方で Inner Loop の改善は Outer Loop からのコントロール下にあることで、要求の変化に対応した情報システムとして利用・運用される。

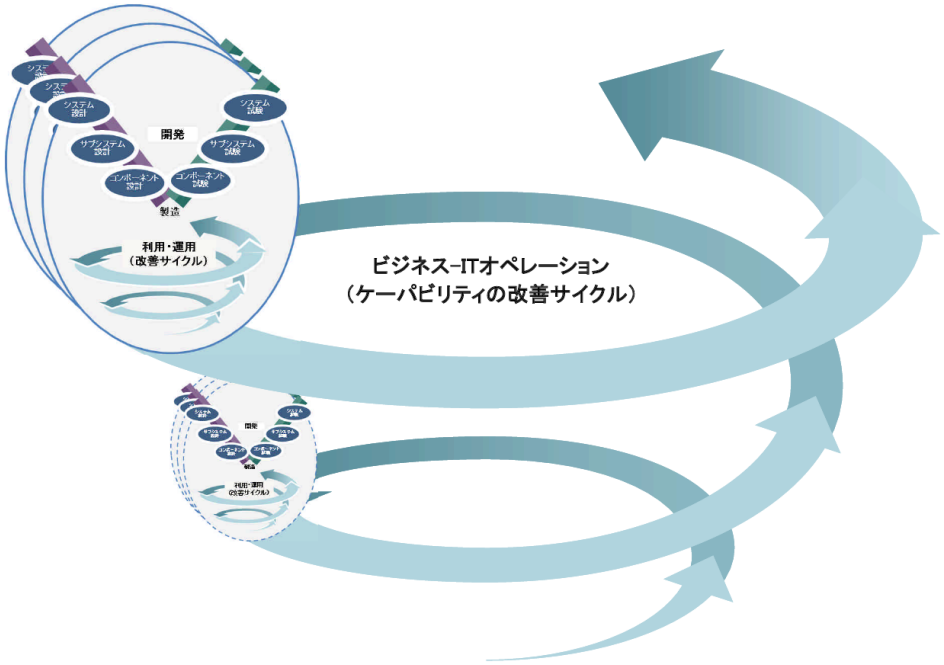
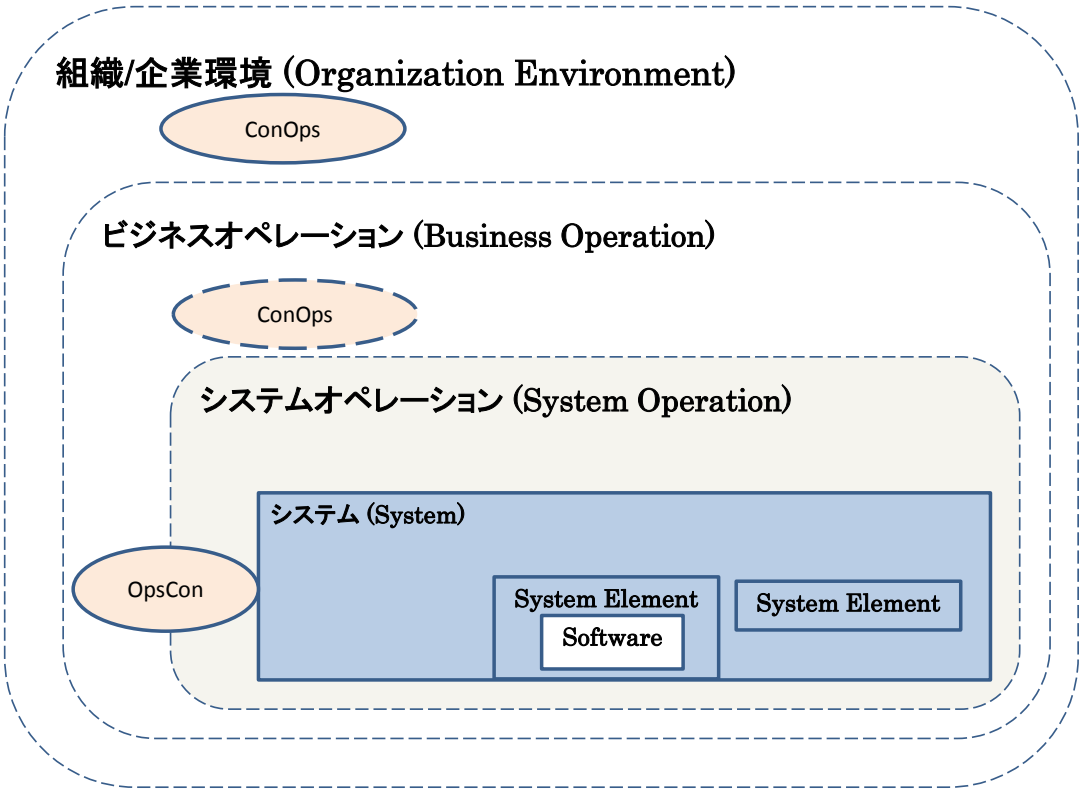


図 35　Outer Loop の改善と Inner Loop の改善

一方で、このようなトップダウン指向での開発対象となるシステムは、ビジネスプロセスのひとつとしてシステムの利用・運用が位置づけられる。全体システムを使った業務に対する組織としての仮説や意図を説明するために ConOps（Concept of Operation）[12]、が描かれ、その下で各システムの OpsCon（Operational Concept）[13]に基づいてシステムが開発され、またシステムが利用・運用される。既存の情報システム開発において情報システムの運用は開発の後段に位置付けられることが多かったが、要求の変化に対応するためにはイメージ的には順序が逆となり、ビジネス運用の一部としてのシステムの利用・運用が位置づけられ、それを満足するためのシステムを構築することになる（図 36）。ITIL ではシステムはあくまで調達するものとして位置づけられているが、それに近い考え方となる。

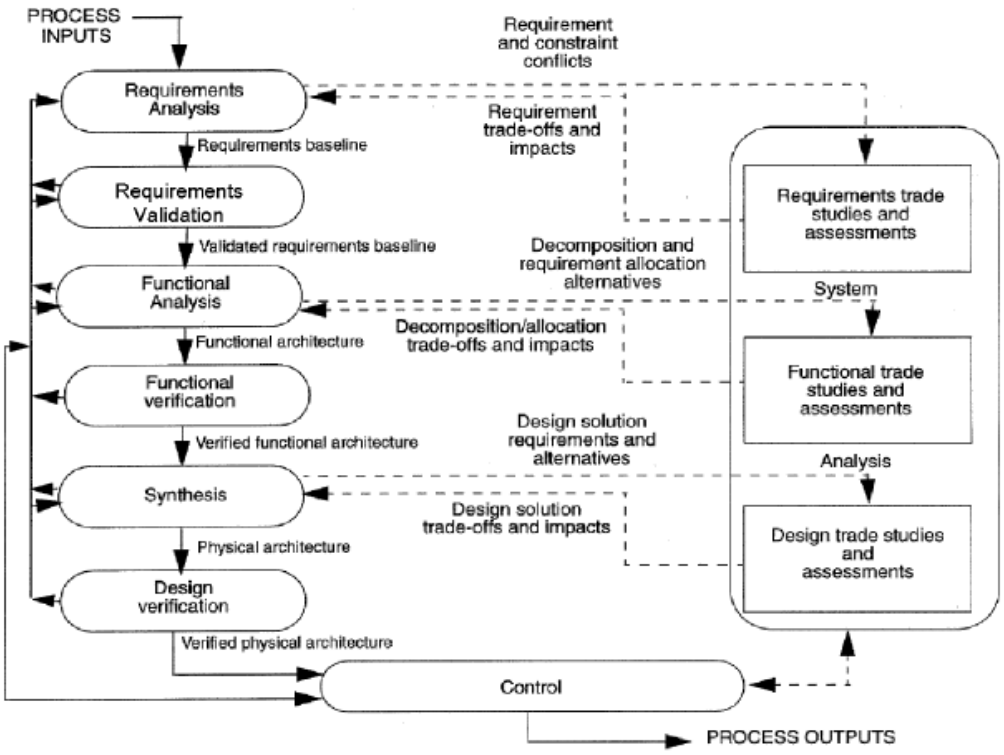
12 ISO/IEC/IEEE29148:2011([19])の Annex B　Concept of operations を参照。
13 ISO/IEC/IEEE29148:2011([19])の Annex A　System operational concept を参照。



ISO/IEC/IEEE29148([19])を参考に作成

図 36　要件定義プロセスの一連の流れの例

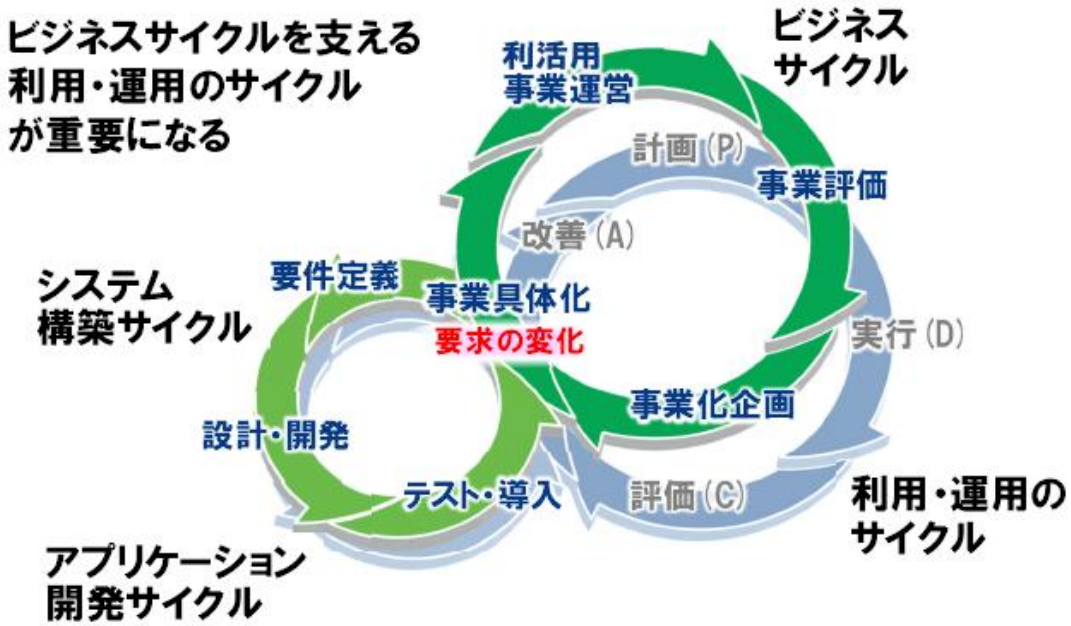
一方で、ビジネス IT オペレーションでは、システム開発側に運用要求が供給されるが、運用要求と実装の関係は留意が必要である。要求分析からアーキテクチャ設計時に ISO/IEC26702（[20], 図 36）に従って考えてみると（詳細は文献[12]を参考にされたい）、要求分析結果と機能分析、アーキテクチャ設計結果は併せて、システム分析を実施する。リスクがないことが確認されなかった場合は、場合によっては要求分析プロセスに戻ることになる。この場合は本報告書での超上流工程部分に運用要求を見直すことになる。アーキテクチャ設計が満足されるまで、運用要求も修正となる可能性があり、サービス開発側からシステムを調達するというよりも詳細なところでは相互の調整が必要になる可能性があることに留意する。



出典：ISO/IEC26702(IEEE1220-2005)[20]

図 37　システムズエンジニアリングプロセス

運用要求はシステム要求の一部に反映され、システム要求の一部は運用要求に影響していると考える（図 38 にも関係例）。



成瀬 WG 委員提供資料（富士通「"FUJITSU SI"のコンセプト　2win cycle model」）を基に一部修正

図 38　ITIL の IT サービスサイクルとシステム構築サイクルの関係例

(2) 前項までのまとめ

冒頭で述べたとおり、要求の変化に対応する技術として、超上流工程の技術を含むビジネス-IT オペレーションのための技術が重要であるという仮説が考えられた。それに対する前項までの考察をまとめると次のとおり。

- 超上流工程の技術力強化が、IT 技術による産業力強化に貢献できる。
- 運用技術の技術課題「継続的改善」は超上流工程と一体化できる可能性がある。
- ビジネス-IT オペレーションとしてまとめるためには、運用技術の技術課題「継続的改善」は超上流工程と上流工程を中心とした開発プロセスをつなぐ役割を期待できる。

（以上、超上流工程プロセスによる分析より）

- アーキテクチャ開発手法のプロセスは、それ自身全体体系として存在する。新しい方法論との適合性・整合性をみながらウォッチする必要がある。

（以上、アーキテクチャ開発手法プロセスによる分析より）

- 開発技術の観点からは上流工程の技術が重要である。このことは信頼性の高い情報システムの構築のケースと同じである。非ウォーターフォール型開発技術はソフトウェア構築に集中している技術と考える。

（以上、開発プロセスによる分析より）

- 他方、ステークホルダーの扱いの観点から、運用について重点を置いていない標準類がみられる。
- PMBOK のようなポートフォリオマネージャ、プログラムマネージャを定義している標準類と、今回洗い出した超上流工程の技術課題との関係性については、検討の余地がある。要求管理者としての位置づけの有無にも留意したい。

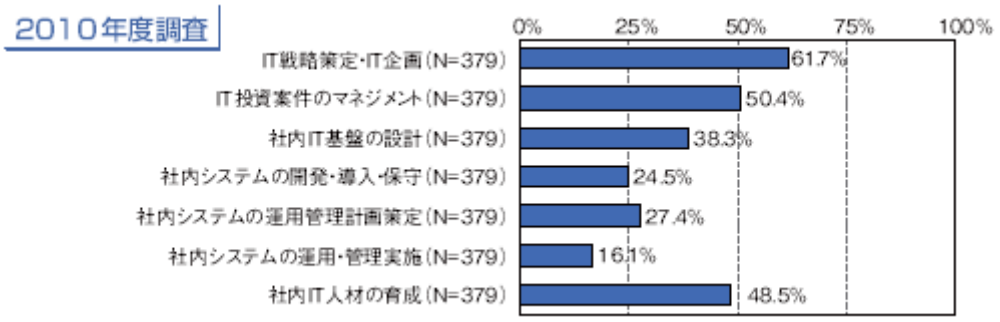
（以上、アーキテクチャ要求による分析より）

なお、最後の点については、それらを統合するためのメタモデルの開発などが対処方法として考えられる。

要求変化を前提とするシステムの開発に関して、超上流工程における技術は、大雑把に言えば、上流工程以降のプロセスでその結果を参照される関係であるが、一方で、超上流工程の技術は、運用プロセスとアーキテクチャ開発手法のプロセスの一部においても重なるもしくは、連携することが可能である。連携可能性が高い ITIL と超上流工程の分析技術の親和性については、実証実験等を通して具体的な方法論としてまとめる必要がある。アーキテクチャ開発手法は非常に大きな技術であるため、本報告書で取り上げた Outer loop の技術との整合性についてはここでは触れていない。十分な検討が必要である。アーキテクチャ開発プロセスについては、あくまでアーキテクチャ開発プロセスが主体であり、Outer loop の考え方を取り込むかどうかということであると推察する。

(3) 技術の訴求について

他の技術も汎用的に重要な技術があるので、大きく異なるとまでは言及しないが、超上流工程、上流工程に力点を置くことが求められる。一方 IPA 人材育成本部「IT 人材白書 2011」によるとユーザ企業が強化すべき業務として、超上流工程に注目していることがわかる。また、運用・管理計画業務や社内 IT 基盤の設計など上流工程に関わる業務を自社業務として強化する方向感がある（図 39）。

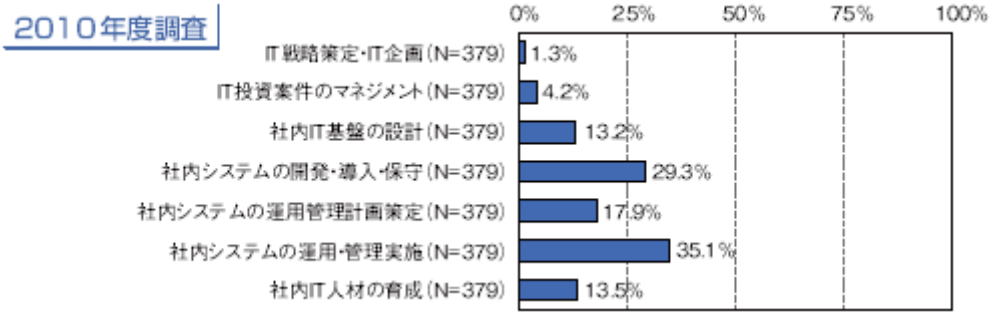


出典： 文献[22]

図 39　今後自社にて強化すべき業務（ユーザ企業）

このことは要求の変化に対応するトップダウン指向であることにフィットしており、コントロール型のシステム構築を受け入れる環境が整いつつあると解釈できる。本アンケート結果からは、ユーザ企業は今後いわゆる「丸投げ」的体質が低下する傾向となる可能性があり、超上流工程、上流工程の技術力を強化することは、重要と思われる。

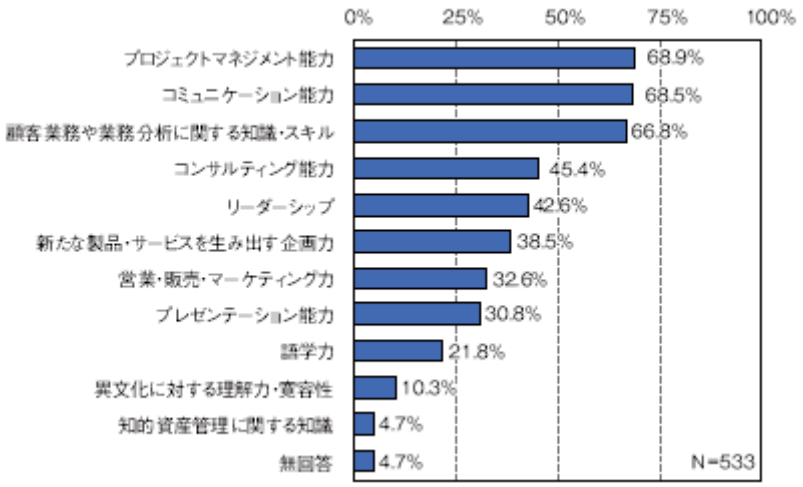
ユーザ企業では開発・導入・保守については子会社等へ期待する技術に移る傾向がある（図 40）。IT 企業側ではクラウドサービスの提供をはじめとして多様なビジネスに展開していくことにマッチしている。アーキテクチャ用語に従うと、品質の高いコンポーネントの提供は IT 企業側に期待することであると推察する。



出典： 文献[22]

図 40　子会社等外部への委託を進めたい業務（ユーザー企業）

IT 企業では、「顧客業務や業務分析に関する知識・スキル」に重要と考えており（図 41）、ユーザ企業との分業点については検討の余地がある。



出典： 文献[22]

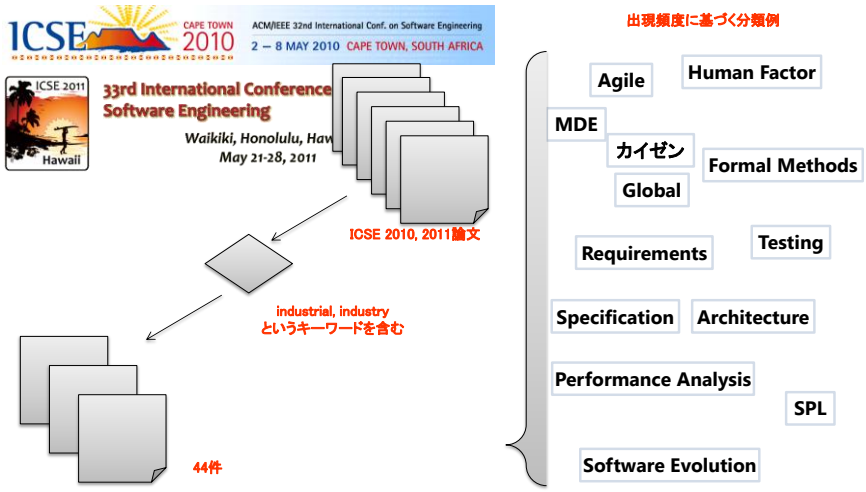
図 41 今後自社の IT 人材にとって重要となるスキル（IT 企業、技術面以外）

(4) ソフトウェア工学分野の技術課題との比較

今回洗い出した技術課題・技術項目をソフトウェア工学分野のトレンド比較するために、ICSE2010,ICSE 2011 から抽出（図 42）できる技術課題とそれらの技術課題を比較した。主な技術課題はのとおりである。

単純に次のことがわかる。

- 超上流分野について技術分野ではない関係もあり、触れることが少ない。この分野での展開が必要と思われる。
- 運用技術と開発技術を関係づける考え方はソフトウェア工学分野としては欠如していると思われる。WG での議論では、論文になりにくい分野であるのが理由であろうとのことである。科学的なアプローチをしているため、推進をすることが重要ではないかと思われる。



出典：http://www.sbs.co.za/ICSE2010/, http://2011.icse-conferences.org/

図 42　ICSE2009, 2010 の技術課題の抽出方法

表 8　ICSE2009, 2010 にみられる技術課題

キーワード	技術課題
Testing	parameterized unit testing, software test automation, combinatorial test design, fault-proneness of aspect-oriented programs
MDE	lacks support for the modeling of architectural design rules
Architecture	architecture-based analysis, creating a systematic approach to designing domain specific distributed, real-time and embedded (DRE) software from software architectural design patterns
Software Evolution	software evolution analysis, impact analysis, change impact analysis, variability analysis, systematic change management, automated traceability, dynamic reconfiguration
Performance Analysis	performance prediction , performance modeling
グローバル	(多拠点での開発用の)assessment framework, cooperation across different countries, times zones, and cultures.
SPL	Symbolic model checking of software product lines
Human Factor	Personality factor conscientiousness on the effectiveness of Pair Programming, self-organizing teams, understanding ″people″ factors, factors that impact the integration experience of newcomers
カイゼン、見積	process simulation, hybrid estimation(HyDEEP), maximum likelihood estimation

5. まとめと課題

要求の変化に対応する情報システム構築技術の適用性関して調査を実施した。2010年度 IPA/SEC が実施した「保守性に高い情報システムの構築技術に関する調査」の結果を、要求の変化という観点で見直し、併せて技術課題をまとめた。2010 年度は、いわゆるシステム要求分析以降のモノ作りの核となる技術課題を洗い出したのに対して、2011 年度は 2010 年度の未実施だった非ウォーターフォール型開発技術に加え、要求の変化をビジネスも含めた視点で分析する超上流工程での要求の変化に資する技術、また要求が変化することから利用・運用に関わる視点での技術課題を加えた結果となった。

その結果を受けて整理軸をプロセスに資するもの、ステークホルダー要求に資するものに分けて検討した。結果として前者は特徴をある程度表現できたのに対して、後者は事例に対する例としての表現にとどまることがわかった。しかし、一方でステークホルダーというキーワードが各種標準類、知識体系で異なる扱いをされていることがわかり、それらを統一して鳥瞰する結果も得ることができ、複数標準類・知識体系を活用する際の指針をみつけることができた。

以下、本調査で得られた課題を列挙する。本課題の解決をすることで、当該技術の訴求に資することができると考える。

① 超上流工程の活動について

- ビジネス-IT オペレーションと本報告書で呼んだ、要求の変化に関係する超上流を中心とした活動（outer loop と呼んだ）は、非常に難しい技術であるが、それ以前にその在り方が課題である。
- つまり、活動の位置づけがビジネスからシステムまで及ぶために、ある意味企業の共通部門的な活動位置を設ける必要がある。さらに検討が必要である。
- さらに変化する要求を管理者もしくは管理するシステムの位置づけも同様な問題となる。
- 一方でケーパビリティ分析を訴求することで、実現した情報システムと組織能力とのミスマッチを早急に避けるようにすべきである。
- 当該技術を担う技術者の育成が求められる。
- PMBOK のポートフォリオマネージャやプログラムマネージャの BOK と当該技術との関係性を明らかにする必要がある。

② 運用技術と超上流技術

- 運用技術の継続的改善は超上流工程と開発工程を結ぶ技術として位置づけることが可能である。つなぐのが役目であるので適切と考えるが、一方で、

どのようにして具体的な情報のやりとりをするというモデルについては今後の検討課題である。

- 冒頭で述べた超上流工程からのトップダウン的に情報システムの継続性と発展をコントロールするという仮説は、運用技術によりボトムアップアプローチでの情報システム状況のモニタリング情報を超上流工程へ渡すことができることを確認したことで妥当性が得られたと考える。従って、トップダウンかボトムアップかの選択ではなく、両者の統合が重要になる。
- ビジネスも含む運用要求の分析と運用実施の両面ができる技術者育成が求められる。

③ 要求の変化と様々な階層性について

- 2.2.1(2-2)のステークホルダー要求の解説の際に掲載した KIKUMA RING の階層性や、TOGAF のビジネスアーキテクチャ、情報アーキテクチャ、技術アーキテクチャ階層がある。このような階層性がなぜ必要とされるか、それらがどのように異なり、どのような共通性があるかを今後明らかにする必要がある。
- また，論文「Israel Navarro, Nancy Leveson, Kristina Lunqvist, Semantic decoupling: reducing the impact of requirements change, Requirements Engineering, Springer, 03 June 2010 」にみられる管理、目的、設計原則、アーキテクチャ、設計仕様、物理仕様、運用という 7 階層に対して 、サービス環境、サービス、運用者、サービス、コンポーネント、サービス確認でどのような「サービス要求水準」が必要になるかを次のような表で整理している。このような表で要求変化を特定し、影響範囲を限定できると考えているようである。どこで変化が発生するかを識別するために，このような階層性が利用できる。

表 9　要求の変化に関する階層性

水準	分類	サービス環境	サービス運用者	サービスコンポネント	サービス確認
0	管理		管理計画		点検結果
1	目的	前提，制約	責任分担	要求，制約	充足性評価
2	設計原則	インタフェース	作業分担	機能分担	分担性評価
3	アーキテクチャ	環境モデル	運用モデル	機能モデル	モデル確認
4	設計仕様	I/F設計	HCI設計	サービス仕様	サービステスト
5	物理仕様		物理設計	コード	テスト
6	運用	監査手順	運用手順	障害報告	運用性確認

Israel Navarro, Nancy Leveson, Kristina Lunqvist: Semantic decoupling: reducing the impact of requirements change, Requirements Engineering, Springer, 03 June 2010.
の表を一部修正

- 変化対応技術ごとに、前提とする階層性が異なる可能性があり、今後，変化対応階層のメタモデルを明らかにするとともに、相互関係を整理することにより、変化対応技術の統合化が期待される。

④ その他

- システムズエンジニアリングの技術に限定すると、宇宙・軍事関係の事例に基づくものを多く採用している。我が国では民生関係にどう適用していくのか、コスト面、人材面を含めて検討が必要である。
- ビジネス-IT オペレーション視点での全体活動の事例の収集が望まれる。
- 本技術課題は部分的には情報システム構築のための実践活動が含まれていると考える。
- システムズエンジニアリングの事例を民生用情報システム開発に置き直し、また一方でTOGAFや当該テーマに合う運用が為されているITIL V3等についても含めて全体事例としてまとめ、さらにその相互関係を明らかにすることで、当該技術を訴求することが望まれる。

以上

参考文献

（非ウォーターフォール関係は文章中に記述）

[1] George Rebovich, Jr. and Brian E. White: ENTERPRISE SYSTEMS ENGINEERING Advances in the Theory and Practice, CRC Press, 2010.

[2] Nancy Leveson: Engineering a Safer World Systems Thinking Applied to Safety, The MIT Press, 2011.

[3] INCOSE: Systems Engineering - Systems Engineering Handbook, 2010.

[4] INCOSE： The Guide to the Systems Engineering Body of Knowledge (SEBOK) Version 0.5 http://www.sebokwiki.org/index.php/Guide_to_the_Systems_Engineering_Body_of_Knowledge_%28SEBoK%29_v._0.5

[5] itSMF（特定非営利活動法人 IT サービスマネジメントフォーラムジャパン）のホームページ（http://www.itsmf-japan.org/itil/）

[6] itSMF Japan コンファレンス富士通講演資料

[7] グローバル情報社会研究所： TOGAF Version 9 －日本語訳版―, 2009.

[8] IPA/SEC： 高信頼化ソフトウェアのための開発手法ガイドブック, 2011.

[9] IPA/SEC： 非機能要求記述ガイド, 2008.

[10] 日本 IBM： 「IBM Global CEO Study 2010 Japan Report」, 2010.

[11] DoD Dupty Chief Information Officer： DoD Architecture Framework Verson 2.02 (http://cio-nii.www.defense.gov/sites/dodaf20/index.html)

[12] IPA/SEC： 非機能要求とアーキテクチャ分析 WG 報告書, 2011.

[13]DEOS White Paper V2.0, www.dependable-os.net/ja/topics/20101201/White_Paper_V2.0J.pdf

[14] IBM: Patterns for e-business（http://www.ibm.com） https://www.ibm.com/developerworks/patterns/

[15] ISO/IEC12207:2008, Systems and software engineering - Software life cycle processes.

[16] ISO/IEC15288:2008, Systems and software engineering - System life cycle processes.

[17] ISO/IEC20000-1:2011, ISO/IEC20000-2:2012 Information technology – Service management

[18] ISO/IEC25010:2011 Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models
[19] ISO/IEC/IEEE29148:2011 Systems and software engineering – Life cycle processes – Requirements engineering
[20] ISO/IEC26702:2007 Systems engineering – Application and management of the systems engineering process
[21] JIS X 0160:2012　ソフトウェアライフサイクルプロセス.
[22] IPA 人材育成本部：「IT 人材白書 2011」, 2011.

謝辞

IPA/SEC　要求発展型開発 WG の委員の皆様には、非常に有益なご示唆を多く頂いた次第である。この場をお借りして御礼申し上げる。

付録 A　米国国防総省(DoD)のケーパビリティアプローチ

（出典：　防衛研究所紀要第 11 巻第 3 号（2009 年 3 月））

システムズエンジニアリングのケーパビリティの例として米国国防省のものをここで取り上げることにする。内容は次のとおりである。

(1) 問題点の所在

冷戦時の脅威は明確に特定されており、シナリオの数も限られていた。しかし冷戦後、脅威が多様化し、不透明感が増大したため、これまでのように脅威を特定し、数少ないシナリオを取り上げて検討するだけでは不十分である状況であった。

(2) シナリオ検討のあり方

これまでのシナリオは、数少ないシナリオを呼称で分類している一次元のシナリオリスト（Name Level Scenario）であった。しかしながら、脅威の多様化などにより、一次元のシナリオリストだけでは十分にシナリオの内容を把握できない。そこで、考えられうる懸念（plausible worries）をすべて一次元のシナリオリストに整理した上で、次のような観点から、多次元のシナリオ空間（Scenario Space）に位置づける。

- 政治軍事的側面（どのように戦争が始まるのか、どの国が同盟関係にあるか等）
- 目標と戦略（目指すべき政治的・軍事的目標は何か）
- 戦力（戦力の規模、特徴、名目上の能力）
- 戦力の効率性（錬度、士気等）
- 環境条件（地形、気候等）
- その他の前提（部隊の機動力等）

(3) 分析のあり方

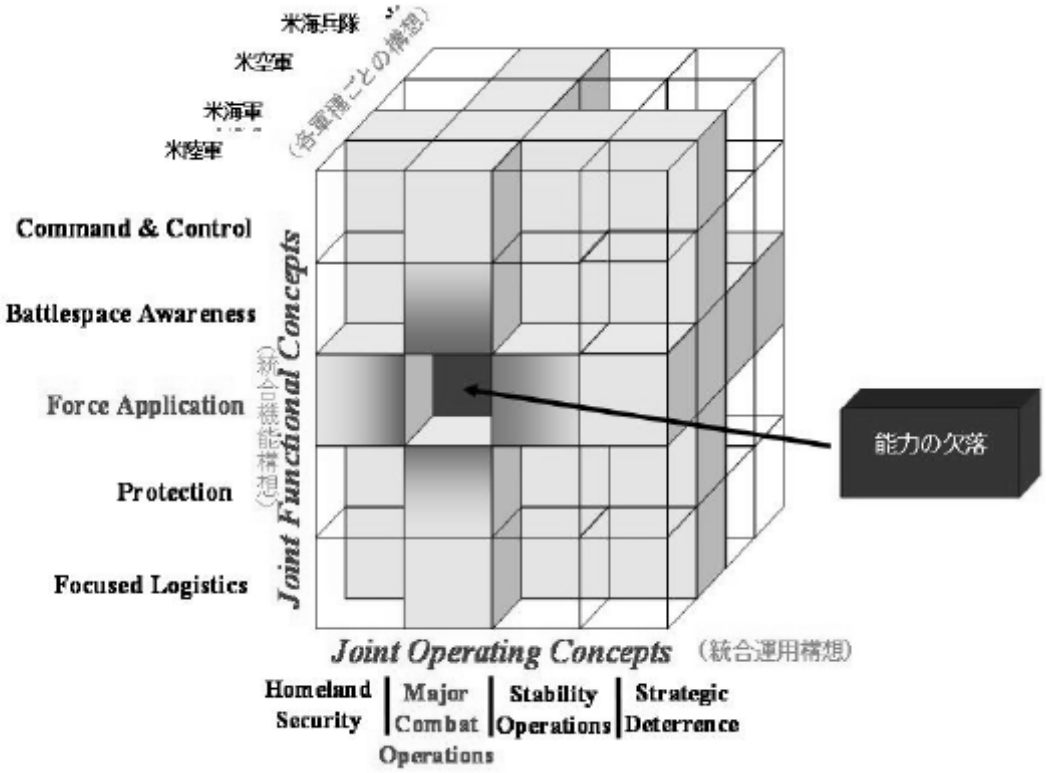
シナリオ空間から、必要となる任務要求（Mission Needs）を洗い出し、その任務要求を満たすケーパビリティの選択肢についてシミュレーションを用いた探求的分析（exploratory analysis）により分析し、評価する。

このケーパビリティベースアプローチに関する考え方は、ランド研究所のポール・デイビス（Paul Davis）により整理されたものである。

Paul K. Davis: Analytic Architecture for Capabilities-Based Planning
Mission-System Analysis, and Transformation, RAND, 2002, pp.23-25. .

(4) 計画のあり方

- ソフトウェアの部品のように、各種軍事能力を部品化し、積木(Building Block)状に組み立てる。それはちょうど下図のようなイメージである。そして、これら各種軍事能力の部品を事態対処に最適な形に組み合わせ、あらゆる事態に対処可能なケーパビリティを経済的に確保する。
- 部品とは、システムズエンジニアリングの分野でオブジェクト指向分析技術を使用して作成されるプログラムの部品を想定していると思われる。



出典： 防衛研究所紀要第 11 巻第 3 号（2009 年 3 月）

図 軍事能力（ケーパビリティ）の部品化のイメージ

付録 B　整理軸に沿った技術課題の整理

（1）　プロセスを中心とした整理

開発工程、超上流工程、運用工程、アーキテクチャフレームワークに分け、それぞれの工程内でのプロセスを整理し、整理軸として、各技術課題をプロット、整理を行う。

① 超上流工程

要求の変化・発展を限定し、Enterprise Systems Engineering のプロセスを活用し、整理を行う。

技術課題 ＼ プロセス			Enterprise Systems Engineering プロセス				
			戦略的技術計画 Strategic Technical Planning	ケーパビリティに基づく開発の分析 Capability-Based Planning Analysis	技術と標準動向調査 Technology and Starndards Planning	エンタプライズ有効性評価 Enterprise Evaluation and Assessment	ステークホルダ分析 Stakeholder analysis
超上流工程分析・評価	システムシンキング		○	○	○	○	○
	ケーパビリティに基づく開発の分析			○			
	エンタプライズ有効性評価					○	
	戦略的技術計画		○				
	技術および標準化動向調査				○		
	ステークホルダ分析						○
	ConOpsの分析/定義		○	○	○	○	○
上流工程における横断的連携	モデリング、シミュレーション、プロトタイピング		○	○		○	○
	機能に基づくSE手法活用						
	オブジェクト指向SE手法活用						
	上流工程開発力強化						
	ロバストネスの確保						
要求管理	要求の明確化						○
	スコープ決め						○
	トレーサビリティ管理/ベースライン管理						○
	コントロールケースの活用						○
システム構成関連技術（部品化・連携）	構造の明確化				○		
	部品化				○		
	クラウド活用				○		
安全と情報セキュリティ	テスト網羅性の保証						
	不測の管理、仮定の管理						○
	運用性、可用性、アシュアランスの評価				○		
	共通基盤としての保証						
	他者に求められる保証						
運用技術	ITIL継続的改善		○	○	○	○	○
非ウォータフォール型開発技術	要求の明確化	機能仕様の定義					
		パフォーマンス仕様の定義					
		プロトタイピング					
		実現可能性調査					
		ビジネス調査					
	構成管理	コーディング規約					
		所有権管理					
		トレーサビリティ管理					
		継続的統合					
	短期間での開発	機能毎の開発					
		コード自動生成					
		シミュレーション					
		テスト駆動開発					
		シンプルな設計のための技術					
	反復による開発	リファクタリング					
		イテレーション計画					
		スプリントレビュー					
		頻繁なリリース管理					
	品質管理	ソフトウェアインスペクション					
		ペアプログラミング					

② 開発工程

開発プロセス標準 ISO/IEC 12207 と ISO/IEC 15288 プロセスを活用し、整理を行う。

技術課題 ＼ プロセス			●:ISO/IEC12207+15288, ○ISO/IEC12207, ■: ISO/IEC15288																						
			● 利害関係者要求定義	● システム要求分析	● システム方式設計	● システム実装	○ ソフトウェア実装	○ ソフトウェア要求事項分析	○ ソフトウェア方式設計	○ ソフトウェア詳細設計	○ ソフトウェア構築	○ ソフトウェア結合	○ ソフトウェア適格性確認テスト	○ ソフトウェア導入	○ ソフトウェア受入れ支援	○ ソフトウェア運用	○ ソフトウェア保守	○ ソフトウェア廃棄	● システム結合	■ システム検証	■ システム移行	■ システム適格性確認	■ システム運用	■ システム保守	■ システム廃棄
超上流工程分析・評価	システムシンキング		○	○	○																				
	ケーパビリティに基づく開発の分析		○																						
	エンタプライズ有効性評価		○	○	○																				
	戦略的技術計画		○	○	○	○	○	○	○																
	技術および標準化動向調査		○	○	○	○	○	○	○	○	○	○	○						○	○	○	○			
	ステークホルダ分析		○	○																					
	ConOpsの分析/定義		○	○	○	○	○	○	○																
上流工程における横断的連携	モデリング、シミュレーション、プロトタイピング		○	○	○	○	○	○	○																
	機能に基づくSE手法活用		○	○	○	○	○	○	○	○	○	○	○						○	○	○	○			
	オブジェクト指向SE手法活用		○	○	○	○	○	○	○	○	○	○	○						○	○	○	○			
	上流工程開発力強化			○	○	○	○	○	○																
	ロバストネスの確保		○	○	○	○	○	○	○			○	○						○	○		○			
要求管理	要求の明確化		○	○	○	○	○	○	○																
	スコープ決め		○	○	○	○	○	○	○																
	トレーサビリティ管理＋ベースライン管理		○	○	○	○	○	○	○	○	○														
システム構成関連技術（部品化・連携）	構造の明確化			○	○	○	○	○	○																
	部品化			○	○	○	○	○	○			○	○	○			○	○	○	○	○	○	○	○	○
	クラウド活用		○	○	○	○	○	○	○										○	○	○	○			○
安全と情報セキュリティ	テスト網羅性の保証		○	○	○	○	○	○	○	○	○	○	○				○		○	○	○	○	○		
	不測の管理、仮定の管理		○	○	○	○	○	○	○	○	○														
	運用性、可用性、アシュアランスの評価		○	○	○	○	○	○	○	○	○		○			○		○	○	○	○	○	○		○
	共通基盤としての保証		○	○	○	○	○	○	○				○				○	○		○	○	○	○	○	
	他者に求められる保証		○	○	○	○	○	○	○	○			○			○				○		○	○		
運用技術	ITIL継続的改善		○	○									○	○		○	○	○			○		○	○	○
非ウォータフォール型開発技術	要求の明確化	機能仕様の定義	○	○		○	○	○		○	○														
		パフォーマンス仕様の定義					○	○	○		○														
		プロトタイピング			○	○	○	○	○	○	○														
		実現可能性調査	○	○		○	○	○	○																
		ビジネス調査	○	○		○	○	○	○																
	構成管理	コーディング規約					○				○														
		所有権管理					○				○														
		トレーサビリティ管理					○	○	○	○	○														
		継続的統合					○				○														
	短期間での開発	機能毎の開発					○			○	○														
		コード自動生成					○	○	○	○	○														
		シミュレーション					○	○	○	○	○	○													
		テスト駆動開発					○	○			○														
		シンプルな設計のための技術					○	○	○	○															
	反復による開発	リファクタリング					○				○														
		イテレーション計画					○	○	○	○	○														
		スプリントレビュー					○															○			
		頻繁なリリース管理					○				○														
	品質管理	ソフトウェアインスペクション					○				○														
		ペアプログラミング					○			○	○														

③ 運用工程

ITIL を活用。各段階をプロセスと見立てて整理を行う。

技術課題 ＼ プロセス			ITIL V3 フェーズ				
			継続的サービス改善	サービス・ストラテジ	サービス・デザイン	サービス・トラジション	サービス・オペレーション
超上流工程分析・評価	システムズシンキング		○	○	○		
	ケーパビリティに基づく開発の分析		○				
	エンタプライズ有効性評価		○				
	戦略的技術計画		○				
	技術および標準化動向調査		○				
	ステークホルダ分析		○				
	ConOpsの分析/定義		○	○	○	○	○
上流工程における横断的連携	モデリング、シミュレーション、プロトタイピング						
	機能に基づくSE手法活用						
	オブジェクト指向SE手法活用						
	上流工程開発力強化		○				○
	ロバストネスの確保						
要求管理	要求の明確化		○				○
	スコープ決め		○				
	トレーサビリティ管理＋ベースライン管理						
	コントロールケースの活用		○				○
システム構成関連技術（部品化・連携）	構造の明確化						
	部品化						
	クラウド活用		○				
安全と情報セキュリティ	テスト網羅性の保証						
	不測の管理、仮定の管理		○				○
	運用性、可用性、アシュアランスの評価						
	共通基盤としての保証		○	○			○
	他者に求められる保証		○	○			○
運用技術	ITIL継続的改善		○	○	○	○	○
非ウォータフォール型開発技術	要求の明確化	機能仕様の定義					
		パフォーマンス仕様の定義	○				○
		プロトタイピング	○		○		○
		実現可能性調査					
		ビジネス調査					
	構成管理	コーディング規約					
		所有権管理					
		トレーサビリティ管理					
		継続的統合					
	短期間での開発	機能毎の開発					
		コード自動生成					
		シミュレーション	○		○		○
		テスト駆動開発					
		シンプルな設計のための技術					
	反復による開発	リファクタリング					
		イテレーション計画					
		スプリントレビュー	○		○		○
		頻繁なリリース管理					
	品質管理	ソフトウェアインスペクション					
		ペアプログラミング					

④ アーキテクチャフレームワーク

TOGAF-ADM(アーキテクチャ開発プロセス)の各フェーズをプロセスと見立てて、整理を行う。

技術課題			TOGAF-ADM									
			初期フェーズ	アーキテクチャビジョン	ビジネスアーキテクチャ	情報システムアーキテクチャ	技術アーキテクチャ	ソリューション	移行計画	実装監督	アーキテクチャ変更管理	要求管理
超上流工程分析・評価	システムズシンキング		○	○	○	○	○	○	○	○	○	○
	ケーパビリティに基づく開発の分析		○	○	○	○	○	○	○	○	○	
	エンタプライズ有効性評価		○	○	○	○			○	○	○	
	戦略的技術計画		○	○	○	○	○	○	○	○	○	
	技術および標準化動向調査			○	○	○	○					
	ステークホルダ分析		○	○	○							○
	ConOpsの分析/定義			○	○	○		○			○	
上流工程における横断的連携	モデリング、シミュレーション、プロトタイピング			○	○	○					○	○
	機能に基づくSE手法活用				○	○		○			○	○
	オブジェクト指向SE手法活用				○	○		○			○	○
	上流工程開発力強化					○	○	○				
	ロバストネスの確保					○	○	○				
要求管理	要求の明確化											○
	スコープ決め											○
	トレーサビリティ管理＋ベースライン管理											
	コントロールケースの活用			○	○	○	○	○			○	○
システム構成関連技術（部品化・連携）	構造の明確化					○	○	○				
	部品化					○	○	○				
	クラウド活用		○	○	○	○	○	○	○	○	○	
安全と情報セキュリティ	テスト網羅性の保証							○				
	不測の管理、仮定の管理			○	○						○	
	運用性、可用性、アシュアランスの評価				○	○	○	○			○	○
	共通基盤としての保証		○	○	○	○	○	○	○	○	○	○
	他者に求められる保証			○								
運用技術	ITIL継続的改善				○	○	○	○				○
非ウォータフォール型開発技術	要求の明確化	機能仕様の定義										
		パフォーマンス仕様の定義										○
		プロトタイピング										○
		実現可能性調査										○
		ビジネス調査										○
	構成管理	コーディング規約										
		所有権管理										
		トレーサビリティ管理										
		継続的統合										
	短期間での開発	機能毎の開発										
		コード自動生成										
		シミュレーション										
		テスト駆動開発										
		シンプルな設計のための技術										
	反復による開発	リファクタリング										
		イテレーション計画										
		スプリントレビュー										
		頻繁なリリース管理										
	品質管理	ソフトウェアインスペクション										
		ペアプログラミング										

（２）　ステークホルダー要求を中心とした整理

整理した１１の要求目的とステークホルダーの関係をステークホルダー要求として、各技術課題をプロット、整理を行う。

＜要求目的＞

- 企業の社会的責任（ＣＳＲ向上）
- 業務効率向上（省力化、業務コスト削減）
- 法令順守・業界標準への準拠
- 業務スピード向上（リードタイム短縮等）
- 経営の透明性確保
- 業務品質・精度の向上
- 経営スピード向上
- 製品・サービスの開発、改善
- 市場シェア拡大
- IT インフラ変更への対応
- 顧客の確保、維持（製品保守、データ量対応、最新技術の活用）

ステークホルダー要求				技術課題	レイヤ		山本分類				要求分類					EA	ISO25010システム/ソフトウェア品質特性														
要求分類		ステークホルダー	補足説明		現実世界	情報システム世界	適応	修正	変革	改良	アプリケーション機能	技術機能	ハードウェア機能	アーキテクチャ要求	IT外要求	③適用処理・④技術	機能適合性	性能効率性	互換性	使用性	信頼性	セキュリティ	保守性	移植性	有効性	効率性	満足性	リスク回避性	コンテクストカバレッジ		
①	企業の社会的責任（CSR向上）	経営企画部門	技術戦略を検討してシステムへの適用を計画する。	ESE技術活用戦略 （Strategic Technology Plannning）	○	○										④															
①	企業の社会的責任（CSR向上）	システム部門	要求が変化したシステムを安全に提供するためには、情報セキュリティが確保できる手順が構築できていることが求められる。	品質保証：共通基盤としての保証	○	○	○							○		④						○									
②	法令順守・業界標準への準拠	システム部門	情報セキュリティが確保できることを保証するために、認証などの他者に求められる保証が必要となることがある。	品質保証：他者に求められる保証	○	○	○				○	○	○	○	○	④						○									
②	法令順守・業界標準への準拠	サービス提供企業	環境の変化から要求は絶えず進化するため、不測の事態を考慮し、ソフトウェアを進化させる必要がある。	要求管理：不測事態への対応、仮定の管理	○		○							○	○	①					○								○		
③	経営の透明性確保	経営企画部門	ステークホルダー要求の変化に対応した開発を許すことができるシステムを導入するために、そのシステムに必要とする能力を事前に把握するために全体プログラムの分析を行う。	ESE ケーパビリティに基づく開発に関する分析 （Capability-Based Engineering Analysis）	○	○										①															
④	経営スピード向上 顧客の確保、維持	システム部門	Functional Analysis/Allocation を行う。入力としては、 Functional Requirements Performance Requirements Program decision Requirements Specifications and Standards Requirements Architectural Concept ConOps Constraints であり、結果として振舞、コンテキスト、コントロールフロー、データフロー、データディクショナリ、ER、Functional Flow Block、Models、Simulation Results、Integrated Definition for Functional Modelingの各図を得る。	上流工程 機能に基づくSE手法活用 Functions-Based Systems Engineering Method	○	○	○	○	○	○	○		○	○		③	○	○					○								○	
⑤	市場シェア拡大	システム部門	情報システムの保守性を良くするためには、ソフトウェアを分解可能な形で部品化しておく必要があり、そのためにはアーキテクチャ記述言語を活用して、構造の明確化と定義を行い、アーキテクチャを明確にする必要がある。	システム構成：構造の明確化		○	○	○	○	○	○			○		④							○	○							
⑤	市場シェア拡大	サービス提供企業	保守性を高めるために、拡張ポイントを明確にするためにスコープ決めを行う必要がある。	要求管理：スコープ決め		○	○				○					①	○	○													
⑥	経営スピード向上 顧客の確保、維持	システム部門	システムズエンジニアリングのプロセスをOMGが中心となってSysMLで記述する方法論を用意している。リアルタイム系の場合 Marteも活用することがある。なお、アシュアランスについても記述する方法論の研究も進んでいる。アクティビティとしては次のとおり。 Analyze Needs, Define Systems Requirements, Define Logical Architecture, Synthesize Allocated Architectures, Optimize and Evaluation Alternatives, Validate and Verify System	上流工程 オブジェクト指向SE手法活用 Object-Oriented Systems Engineering Method	○	○	○	○	○	○	○		○	○		③	○	○					○								○	
⑥	経営スピード向上 顧客の確保、維持	サービス提供企業	情報システムを保守し、要求を明確化し、見えるような状態にしておき、変化に耐えるようにする必要がある。	要求管理：要求の明確化	○	○	○	○	○	○																					
⑥	経営スピード向上 顧客の確保、維持	サービス主管部門	顧客のニーズに従った商品を適時に提供するため、システムを絶えず進化させるための前提として要求の変化を明確化する必要がある。そのため、要求を明確化する技術を必要とする。	要求の明確化	○	○	○	○	○	○	○			○		③	○														
⑦	業務効率向上（省力化、業務コスト削減）	システム部門	環境の違い等の少々の変化による改修を避けるために、冗長性を確保したソフトウェアの実現を確認するロバストネステストを実施する。	テスト技術：ロバストネスの確保		○	○		○		○			○		③	○		○				○	○							
⑧	業務スピード向上（リードタイム短縮等）	経営企画部門	システム全体のもつべき全体の統合の在り方を分析し、そのコンポーネントとアーキテクチャ構成を診断する。	ESE システムのあるべき全体統合の調査・分析 （Enterprise Analysis and Asssesment）	○	○										①															
⑨	業務品質・精度の向上	経営企画部門	当該システムのステークホルダーが誰であり、どのような役割をもっているのかを分析する。	ESEステークホルダー分析 （Stakeholder analysis）	○	○										①															
⑨	業務品質・精度の向上	システム部門	システムズエンジニアリングに限らず、ソフトウェアエンジニアリングでも重要な技術である。変化した要求への対応について上流工程段階で設計検証することは非常に重要である。基本的な手法として位置づける。	上流工程 モデリング、シミュレーション、プロトタイピング Modeling, Simulation, Prototyping	○	○	○	○	○	○	○	○	○	○		③	○	○					○								
⑩	製品・サービスの開発、改善	経営企画部門	システム全体としての Operation の在り方と、それに従ったシステムのコンポーネントのOperationの在り方、相互依存関係などを分析して、システムのOperation の概念をまとめる。	ESE ConOpsの分析/定義	○											①															
⑪	ITインフラ変更への対応（製品保守、データ量対応、最新技術の活用）	経営企画部門	今後の技術動向を予測する。	ESE技術および標準動向調査 （Technology planning）		○										④															
⑪	ITインフラ変更への対応（製品保守、データ量対応、最新技術の活用）	システム部門	データ量の変化や、ハード・ミドル変更の対応、OS変更の対応など将来変わる可能性がある。	コントロールケースによる非機能要求記述	○	○	○		○					○		①					○		○	○					○		

（3） 品質特性を中心とした整理

ISO25010 システム/ソフトウェア品質特性に対して、各技術課題をプロット、整理を行う。

技術課題	ステークホルダー要求			要求分類					EA	ISO25010システム/ソフトウェア品質特性												
	要求	ステークホルダー	補足説明	アプリケーション機能	技術機能	ハードウェア機能	アーキテクチャ要求	IT外要求	③適用処理・④技術	機能適合性	性能効率性	互換性	使用性	信頼性	セキュリティ	保守性	移植性	有効性	効率性	満足性	リスク回避性	コンテクストカバレッジ
システムシンキング	モデリングを活用した迅速な開発	システム部門	システムズエンジニアリング手法を使う場合、システムシンキングが前提となっている。そのなかで迅速な開発でかつ説明責任を果たせるモデリングのためにはシステムシンキングが必須である。						①					○			○	○				○	
ケーパビリティに基づく開発の分析（Capability-Based Engineering Analysis）	経営の透明性確保	経営企画部門	ステークホルダー要求の変化に対応した開発を許すことができるシステムを導入するために、そのシステムに必要とする能力を事前に把握するために全体プログラムの分析を行う。						①	○	○	○	○	○	○	○	○	○	○	○	○	
エンタプライズ有効性評価（Enterprise Analysis and Assesment）	業務スピード向上（リードタイム短縮等）	経営企画部門	システム全体のもつべき全体の統合の在り方を分析し、そのコンポーネントとアーキテクチャ構成を診断する。						①		○			○				○			○	
戦略的技術計画（Strategic Technology Planning）	企業の社会的責任（CSR向上）	経営企画部門	技術戦略を検討してシステムへの適用を計画する。						④	○	○		○		○				○	○	○	
技術および標準動向調査（Technology planning）	ITインフラ変更への対応（製品保守、データ量対応、最新技術の活用）	経営企画部門	今後の技術動向を予測する。						④		○		○		○						○	
ステークホルダー分析（Stakeholder analysis）	業務品質・精度の向上	経営企画部門	当該システムのステークホルダーが誰であり、どのような役割をもっているのかを分析する。						①	○						○		○		○	○	
CONOPSの分析/定義	製品・サービスの開発、改善	経営企画部門	システム全体としての Operation の在り方と、それに従ったシステムのコンポーネントのOperationの在り方、相互依存関係などを分析して、システムのOperation の概念をまとめる。						①	○			○	○	○	○		○		○	○	
モデリング、シミュレーション、プロトタイピング	業務品質・精度の向上	システム部門	システムズエンジニアリングに限らず、ソフトウェアエンジニアリングでも重要な技術である。変化した要求への対応について上流工程段階で設計検証することは非常に重要である。基本的な手法として位置づける。	○	○	○	○		③	○	○					○						
機能に基づくSE手法活用	経営スピード向上 顧客の確保、維持	システム部門	Functional Analysis/Allocation を行う。入力としては、 Functional Requirements Performance Requirements Program decision Requirements Specifications and Standards Requirements Architectural Concept ConOps Constraints であり、結果として振舞、コンテキスト、コントロールフロー、データフロー、データディクショナリ、ER、Functional Flow Block、Models、Simulation Results、Integrated Definition for Functional Modelingの各図を得る。	○		○	○		③	○	○					○						
オブジェクト指向SE手法活用	経営スピード向上 顧客の確保、維持	システム部門	システムズエンジニアリングのプロセスをOMGが中心となってSysMLで記述する方法論を用意している。リアルタイム系の場合 Marteも活用することがある。なお、アシュアランスについても記述する方法論の研究も進んでいる。アクティビティとしては次のとおり。 Analyze Needs, Define Systems Requirements, Define Logical Architecture, Synthesize Allocated Architectures, Optimize and Evaluation Alternatives, Validate and Verify System	○		○	○		③	○	○					○						
上流工程開発力強化	迅速な開発、顧客確保、維持	システム部門	顧客名簿を失わない用に迅速な開発を必要としている場合、W字開発を活用して、テスト技術を上流工程で活用した開発はひとつの方法論である。	○					①	○						○						
ロバストネスの確保	業務効率向上（省力化、業務コスト削減）	システム部門	環境の違い等の少々の変化による改修を避けるために、冗長性を確保したソフトウェアの実現を確認するロバストネステストを実施する。	○			○		③	○		○				○	○					
要求の明確化	経営スピード向上 顧客の確保、維持	サービス提供企業	情報システムを保守し、要求を明確化し、見えるような状態にしておき、変化に耐えるようにする必要がある。	○		○	○	○		○						○					○	
要求の明確化	経営スピード向上 顧客の確保、維持	サービス主管部門	顧客のニーズに従った商品を適時に提供するため、システムを絶えず進化させるための前提として要求の変化を明確化する必要がある。そのため、要求を明確化する技術を必要とする。	○			○		③	○												
スコープ決め	市場シェア拡大	サービス提供企業	保守性を高めるために、拡張ポイントを明確にするためにスコープ決めを行う必要がある。	○					①	○	○											
トレーサビリティ管理/ベースライン管理	経営スピード向上 顧客の確保、維持	システム部門	顧客ニーズに従った商品を適時に提供するためには、システムを適時変化させる必要があり、その状況をしっかりと管理する必要がある。	○	○	○	○	○		○				○	○	○						
コントロールケースによる非機能要求記述	ITインフラ変更への対応（製品保守、データ量対応、最新技術の活用）	システム部門	データ量の変化や、ハード・ミドル変更の対応、OS変更の対応など将来変わる可能性がある。				○		①					○		○	○				○	
構造の明確化	市場シェア拡大	システム部門	情報システムの保守性を良くするためには、ソフトウェアを分解可能な形で部品化しておく必要があり、そのためにはアーキテクチャ記述言語を活用して、構造の明確化と定義を行い、アーキテクチャを明確にする必要がある。	○			○		④							○	○					
部品化	経営スピード向上	システム部門	情報システムの保守性を向上させるために、部品化を行った。コンポーネント化を徹底することでシステム化も図り、ソフトウェア信頼性向上も図っている。	○			○		④	○		○		○		○						○
クラウド活用	経営スピード向上	システム部門	情報システムの保守性・性能効率性を確保する。	○			○		③	○	○			○		○						
テスト網羅性の保証	法令順守・業界標準への準拠	システム部門	情報システム開発においてもポイントのところではテスト網羅性を保証することで信頼性を確保することが求められる。	○				○	③	○				○	○					○		
不測の管理、仮定の管理	法令順守・業界標準への準拠	サービス提供企業	環境の変化から要求は絶えず進化するため、不測の事態を考慮し、ソフトウェアを進化させる必要がある。				○	○	①					○							○	
運用性、可用性、アシュアランスの評価	企業の社会的責任（CSR向上）	システム部門	社会システムへの影響があるシステムについて、運用性、可用性、そしてアシュアランスを確保する必要があり評価が必要である。	○	○	○	○	○	③	○				○	○	○					○	
共通基盤としての保証	企業の社会的責任（CSR向上）	システム部門	要求が変化したシステムを安全に提供するためには、情報セキュリティが確保できる手順が構築できていることが求められる。				○		④						○							
他者に求められる保証	法令順守・業界標準への準拠	システム部門	情報セキュリティが確保できることを保証するために、認証などの他者に求められる保証が必要となることがある。	○	○	○	○	○	④						○							
ITIL継続的改善	経営スピード向上	システム部門	モニタリングを実施し、迅速な業績把握に貢献し、ビジネスおよびシステム、その運用を改善する。	○				○	①									○	○	○	○	
要求の明確化	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	非ウォーターフォール型開発技術により要求を達成するためには、初期の要求を明確化することが必須である。	○					④	○										○		
構成管理	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	非ウォーターフォール型開発技術により要求を達成するためには、構成管理技術が前提となる。	○					④	○										○		
短期間での開発	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	早期改善や精度向上のためには短期間開発技術が有効である。	○					④	○										○		
反復による開発	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	精度向上や改善のために反復による開発のためには反復型開発が有効である。	○					④	○	○		○							○		
品質管理	業務品質・精度の向上 製品・サービスの開発、改善	システム部門	非ウォーターフォール型開発技術の品質向上のために、ペアプログラミング等の方法での開発を適用する。	○					④	○				○		○						