Digital Service Playbook

ver.0.9.0

2020年12月

経済産業省

情報プロジェクト(DX)室

ver.	リリース	概要
0.9.0	2020/12	新規リリース。以下の章はTBAとする。 3.2.3. デプロイ 3.3.1 保守・運用

＜なぜこのPlaybookを作るのか＞

我々経済産業省DX室が目指すのはユーザーにとって利用しやすいデジタルサービスを、効率的、効果的に開発することである。
一方で現状はユーザーに寄り添ったサービスデザインがまだあまりなされておらず、その手法も省内で確立していない。
加えて、ツールやソフトウェアの共有、開発プロセスのベストプラクティスの共有が効率的、効果的になされていない。
良い取組が再現され、さらにその上に良いやり方が生まれるといった状況にならなければ、いつまでも最短距離で良いデジタルサービスを組織として作っていくことはできない。
シェアすることは大きな価値を持つ。シェアを通じて自分が行なっていることが相対化され、組織の「カルチャー」を作っていくことにつながる。これを形にして残していくことで「ルール」が生まれる。
現在の行政組織のカルチャーは古く、これに併せた形ではいつまでも旧来型の「行政システム開発」の「ルール」から抜け出すことはできない。
CIO補佐官、デジタル化推進マネージャーといったエキスパート集団が経産省の中で新しいプラクティスを重ねていくことで、初めて新しい「カルチャー」・「ルール」が生まれ、組織が変革されるとともに国としても良いサービスが提供できるようになるはずだ。
やり方を変え続け、洗練させなければ、良いものは生まれない。現状の組織のあり方に文句を言うのでなく、皆さんが新しいカルチャーを経産省に作るという強い意志を持ってこのプレイブックを発展させていって欲しい。

目次

1.DXオフィスのカルチャー

以下DX室としてどういったマインドセットで働いていってほしいかを記載する。エキスパート集団として、皆でカルチャー・ルールを作っていく上で共有すべき行動規範を示す。

1. セルフスターターであること

- 自分がプロジェクトのオーナーであるとの自覚と責任を持つこと。
- 行政のデジタル化は日本を変革する大事な役割であるという認識の下に、①大局的な視点と②プロジェクト単位の視点の両方を持つ。
- ①の視点は自分が政府の、経済産業省のDXにおいてどんな役割を果たすべきかを客観的に把握できる。
- ②は個別のサービスをコントロールする上で必ず必要。それぞれのプロジェクトにアサインされている行政官、ベンダーの力量を把握し、自分がどこまでの働きをするべきか考える。

2. フラットな立場で議論し、関係者を巻き込むこと

- 職場において上下関係はない、役割の違いがあるだけだと考える。議論の際も、相手の役職にこだわらず課題を議論する。
- 一方で人格的に課題のある人と仕事をすることもあるかもしれない。その際は相手の立場に立ってみて、その人がこだわっている点は何なのかを探り、その点について安心感を与えてあげるように誘導していく。
- また、省内・外部のステークホルダーとの関係構築の感度を高め、アンテナを立てておく。省内の人は部署が変わっても一緒に仕事をした経験などつながりがあれば、支援者になってくれるかもしれない。外部の人は連携することで自分の課題解決の糸口が見つかるかもしれない。
- 課題解決のアイディアと同じくらい、それを実現するために必要な支援者のプールを持つことは重要。

3. 情報をシェアし、助けあうこと

- 自分が学びの中で得たインサイトや、他の人にも知ってもらいたいと思うことを積極的にシェアする。
- シェア内容は①自分の業務の内容、②自分の関心事項の二つがあるが、両方をシェアすること。
- ①のシェアで、同じ課題にぶつかった際の解決速度は上がり、サービスの連携等による付加価値は上がる。
- ②のシェアで、自分の関心分野のフラッグを立てられるだけでなく、新しい技術トレンドやサービスに関する感度を高め、クリエイティビティが増す。
- シェアを通じて、課題にぶつかった時に誰に助けを求めればよいかがわかる。助け合うことでチームとしての一体感が生まれる。いつでも助けを頼める関係性を皆で作っていく。

4. 失敗を恐れずチャレンジすること

- 失敗する、課題が生じるということはそれだけ難しいことにチャレンジしているという証拠であり、これらにぶつからないということは現状に甘んじているということである。
- おかしいと思ったことは戦い、攻めきれるところまで攻める。その一歩が世の中を変えると信じる。
- 組織内の課題解決が必要な時は行政官に助けを求め、2人3脚で突破する。失敗したとしてもサービスはワンショットでは終わらない。そこから学んだことを次のフェーズで反映させれば良い。
- 2.①のシェアは、失敗の経験、そこから何を学べたかも含む。プラクティスを蓄積し、失敗を成功への道につなげる。

5. ワクワクする仕事をすること

- 直面している仕事は辛いものかもしれない。でもそれが成功したら日本はどう変わるだろうか。サービスの対象者はどれだけ楽になるだろうか。その仕事の先の未来にワクワクできるならそれを少しでも楽しむ。
- 我々のイマジネーションがサービスデザインを変え、日本のあり方を変える。自分が信じなかったら、誰が信じるだろうか。
- 世の中の多くの既成概念は一握りの人たちが盲目的に従った結果としてできている。我々はデジタルサービスを提供するとともに、その既成概念をぶち壊す仕事をしている。
- 新しいデジタルインフラを作る仕事をしている。ワクワクしないだろうか。新しい社会をデジタルでデザインしよう。

2.サービス開発の基本的な考え方

本プレイブックでは開発のフェーズを3つに分けて整理する。その基本的な考え方について以下に示す。

1. サービスをデザインする

- まずは現状にどんな課題があり、それをどのようなデジタルサービスで解決するのかといったことを整理し、基本的な設計、デザインを行う必要がある。
- 重要なのは誰にとってのどのような課題を解決するかを特定することである。その結果として目指すべき理想像を明らかにし、そのギャップを明らかにする。
- その上でユーザーの体験をどのようにデザインするかといったサービスデザインの検討が必要である。実際のユーザーの立場に立ち、どのようなペインがあるのかをサービスのジャーニーを追うことで明らかにし、そのペインの解消をどのようなサービスで行うのが望ましいかを検討する。
- その際に、既存のシステム、プロセスが確立されている場合にはそれがデジタルテクノロジーを前提として効率化されているかを見直す必要があり、BPRを通じて、サービスの裏側のオペレーションにおいても不要なデータや、人の介在を最小限にするデザインを考える。
- これらを検討した上でデジタルサービスに求められる要件定義、仕様書作成を行うことで、実現したいサービスの全体像を取りまとめる。仕様書においては、経産省として標準的に利用するツールや、接続する既存サービスなどについても整理しておく。
- また、この段階で実際にサービスの効果を検証する上でのKGI、KPIや費用対効果などについても整理しておくことで、実際にサービスが開始した後の効果検証の方法についてもあらかじめデザインしておく必要がある。

#課題設定　#サービスデザイン　#BPR

2. サービスを形にする

- サービスのデザインができた上で、どのように開発を行うかを検討していく。サービスの性質を考えた上で、既存のツールを利用し内製で開発可能なのか、それともITベンダーへの委託が必要なのかを判断する。
- ITベンダーに委託する際にはそのベンダーの能力を評価するための評価基準を持つ必要があるだけでなく、幅広い市場のプレーヤーについてソーシング可能な環境を持つ必要がある。加えて、ベンダーが特定されたとしても、実際に開発に携わるチームが2人3脚で走れるメンバーなのか、実力があるのかを把握するためにハッカソンなどを通じてその実力を評価する仕掛けも必要である。
- 開発に当たってはアジャイル型の開発手法を取るのか、ウォーターフォール型の手法を取るのか判断する必要がある。
- ユーザーの使いやすさが重視されるフロントエンドのサービス、システムオブエンゲージメント（SoE）の場合は、作りながら、ユーザーテストを行い、反応を見て改善していくアジャイル型の開発が望ましいと考えられるのに対し、データベース等、きちんと仕組みとして安定稼働が求められ、開発方法も確立しているシステムオブレコード（SoR）の場合には、ウォーターフォール型の開発も考えられる。開発の内容、チームの能力に応じて、どのような開発を行うのか選択する必要がある。
- またチームメンバーの役割についてもベンダーだけでなく、経済産業省サイドでも予め割り当て、どのようなフォーメーション、責任分担で開発を進めるのかを整理しておく必要がある。
- サービス開発に当たっては、経産省全体のシステムとの関係でどのように位置付けられるのか、どのような連携の可能性があるのかも念頭に置きながら進める。更に他省庁のシステム

や、民間のデジタルサービスも含め、ソフトウェアエコシステム全体においてユーザーにとって最適な構成を考え、APIでの接続関係をデザインする必要がある。

- 政府・経済産業省で作成したサービスであることを示し、ブランディングを進める上では、デザインシステムの導入等を通じて開発されたサービスの統一感も意識する。
- システムセキュリティや取り扱うデータの機密性の扱いについても配慮しながら構築を進め、テスト工程もなるべく自動化を進めていく必要がある。
- 構築したサービスについてはスモールグループのユーザーによるレビューなどを通じてユーザビリティテストを行い、実際のユーザーにとって使いづらい部分がないかを確認し、リリース前の調整を行う。
- ソフトウェア資産を統一的に管理し、開発したベンダーのみがソースを理解できるといったことが生じないような仕組みを検討し、経産省が運用する統一的なガバナンス体制を構築する。

#要件定義 #調達 #開発 #テスト/デプロイ

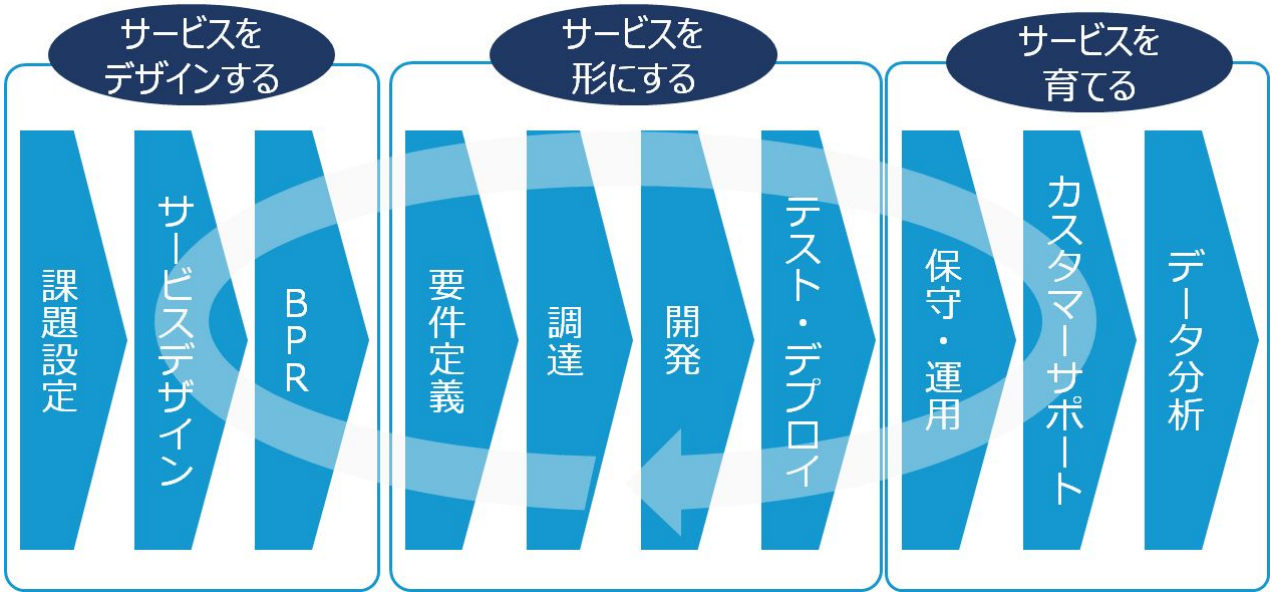
3. サービスを育てる

- DevOpsのコンセプトを持ち込み、開発と運用を一体的に行い、自動化を進めることで、ヒューマンエラーを減らし、安定的なシステム運用と機能の拡充の両方を実現していく必要がある。
- サービスをリリースした後もこれが利用されるように育てていく、より便利にしていく必要がある。既存ユーザーのエンゲージメントを高めるにはユーザーがサービス利用で行き詰まった際に手厚くサポートすることが望まれる。ユーザーが行き詰まった箇所がわかれば、その箇所の機能に課題があることがわかり、サービス改善の糸口にもなりうる。
- 加えて、当初のサービスの目的を達成しているかを評価するため、KGI、KPIの推移を評価していく必要がある。その他ユーザーデータに着目することで、本当に利用しているユーザーは誰で、どの機能を使っているのか、新しい機能へのニーズはないのかを追いながら改善を繰り返していく必要がある。
- 利用率を高めるためには、マーケティングを通じユーザーがアクセスするチャネルで認知を高め、利用へのコンバージョンを高めていくことが望まれる。
- サービスはリリースしたら終わりではなく、継続した改善を通じて真にユーザーに使いやすい状況を追求し続ける姿勢が重要であり、これこそが行政サービスに欠けている姿勢である。
- 分析結果を通じて、再度課題を問い直し、サービスを改善することが重要であり、サービスを「デザインする、形にする、育てる」のプロセスはループして何度も繰り返されるべきものである。

#保守/運用 #カスタマーサポート #データ分析

3. 各フェーズにおけるルール

以下では、開発におけるそれぞれのフェーズにおいて①活用すべきフレームワーク、②共通して活用すべきITツール、③まとめるべきドキュメント、④参照すべき政府全体、経済産業省のルールなどについてまとめる。
これらをCIO補佐官、デジタル化推進マネージャーが参照することで、サービス開発の標準化を進めるとともに、随時見直しを行うことによって、より高い品質のサービスが開発されるプラクティスを蓄積していく。



3.0 フェーズ概要

3.1.1 課題設定

誰のどのような課題を解決するサービスであるべきか、特に行政システムを開発する上での重要な観点についてシェアする。

3.1.2 サービスデザイン

推進する上での思考や進め方のベストプラクティスを紹介する一方、豊富な経験から得られた、成功のためのDO/DON’Tについてシェアする。

3.1.3 BPR(Business Process Re-Engineering)

主に筆者の経産省バックオフィスシステム構築の経験から、BPRがうまく回らない原因と解決への取組みについてシェアする。

3.1.4 要件定義

政府調達という制約のなか、要件定義フェーズで要点として押さえるべきポイント、漏れがちな機能/非機能要件をどうしたら救えるかについてシェアする。

3.2.1 調達

IT関連事業の政府調達手続(公共調達)がどのような背景で現在の形に至ったか、そして検収までの詳細なプロセスについてシェアする。

3.2.2 開発

官公庁システム開発が抱える、民間にはない”単年度事業原則”と”入札制度”という2つの大きな制約がもたらす課題に対する解決方針をシェアする。

3.2.3 テスト・デプロイ

品質保証への責務と、後工程であるが故の期日に対する逼迫感で様々な課題が散在するテスト工程に対して、体制革新や今すぐできる解決策をシェアする。
また行政サービスでのデプロイ時の課題やCI/CD活用の考え方を整理する。

3.3.1 保守・運用

開発と同様、単年度事業原則の中で4月1日を境界としたサービスインと運用開始をシームレスに実現するためのポイントや、運用におけるモニタリング・障害対応の考え方をシェアする。

3.3.2 カスタマーサポート

サポートの利用者と運営者それぞれの立場での課題に対し、時代に沿った解決方針とその重要性をシェアする。

3.3.3 データ分析

サービス稼働後のデータ分析による改善サイクルがうまく定着しない原因と、そのベストプラクティスについてシェアする。

3.1サービスをデザインする

3.1.1課題設定

デジタルサービスを開発する際には、単にシステムを利用者に提供するのではなく、利用者にとって使いやすく、本当に必要とされる「サービス」の提供を意識する必要がある。その一方で、デジタルサービスを通じて、行政機関としてのあるべき役割をどう果たしていくのかというのを意識しておく必要がある。
サービスデザインに向けた課題設定においては、”誰にとってのどのような課題を解決するのか”、また”課題解決に向けてどのようなサービスをどう構築していくのか”という二点を検討する。
本資料では便宜上、前者を「課題発見」、後者を「サービス戦略」と呼称する。

1. 課題発見：解決すべき課題の特定、打ち手の仮説立案
2. サービス戦略：サービスのビジョン・ロードマップ、KPIの検討

1. 課題発見

1) 本フェーズにおける課題

課題発見においては、主に以下のような課題が存在する。

- 現状を正しく理解できていない（1-1）
- 行政サービスとしてのゴールを意識できていない（1-2）
- 課題の深堀りと対応案の検討が不十分（1-3）
- 制約条件を考慮できていない（1-4）

1-1. 現状を正しく理解できていない

- 行政側の思い込みによって、利用者の実情を把握したつもりになっており、的外れな課題を設定してしまう
- 必要な粒度で情報を収集・整理できておらず、現状に対する解像度が低いままになっている
- 全体の一部しかフォーカスできておらず、エンドツーエンドで関連する全てのプロセスを見渡せていない

1-2. 行政サービスとしてのゴールを意識できていない

- 利用者の利便性は改善されたものの、最終的に行政サービスとして意図した成果が実現できていない
- そもそも組織として、該当するサービスを通じて成し遂げたいこと、目指したいゴールが定義されていない

1-3. 課題の深堀りと対応案の検討が不十分

- 課題の根本原因を把握せずに対応案を考えてしまい、的外れなデジタルサービスを構築してしまう
- 本来であれば課題でないもの、もしくは優先度が低いものについても、一度にシステム化しようとしてしまう
- BPRや制度変更などで対応すべき課題についても、不必要にシステム化しようとしてしまい、スコープが膨れあがってしまう

1-4. 制約条件を考慮できていない

- 解決すべき課題に優先度がつけられておらず、全ての課題に対して網羅的に取り組もうとしてしまい、期間・予算内に収まらず破綻してしまう。
- インパクトの小さい課題から取り組んでしまい、目に見える効果がなかなか現れず、取組全体の勢いを削いでしまう
- 課題解決の費用対効果（ROI）やタイムラインを示せておらず、予算要求プロセスや関係者の巻き込みが円滑に進まない

2) 本フェーズの進め方

課題発見は、以下のステップに沿って実施する。

- 現状を正しく理解する（2-1）
- 行政としてのゴールを定義する（2-2）
- 取り組むべきサービス仮説を立案する（2-3）
- 費用対効果をもとに解決すべき課題を絞り込む（2-4）

2-1. 現状を正しく理解する

- 課題発見の第一歩として、対象となる手続きの現状を正確に把握し、現状に対してどのような改善ができるのかを考える。
- 対象手続・業務の必要性が発生した時から完了まで、時系列にユーザーの行動を洗い出し、エンドツーエンドの全体像を整理する。サービスの利用者だけでなく、関連する全ステークホルダー（例：省内の審査担当）を把握し、各主体の行動を理解すること。各主体の作業を5W1Hの粒度で整理すると分かりやすい。

5W1Hのフレームワーク（例）

項目	具体例
誰が（Who）	申請者
何のために（Why）	経産省から〇〇の許可を受領するため
いつ（When）	毎年1回（不定期）
どこで（Where）	窓口
何を（What）	申請書・添付書類の提出
どれくらい （How much）	申請書：50枚、添付書類：5枚

- Whoについては、その特性（ITリテラシーや利用頻度など）に応じて、サービス利用者をグルーピングする。各グループごとにペルソナを定義し、現状に対する不満や行動の目的を整理しておくことが望ましい。また、それぞれのグループの人数感を把握し、サービスを通じてリーチする人数をざっくりと把握しておくこと。
- 現状調査に当たっては、以下のような代表的な手法を用いて、思い込みや勘違いなどのバイアスを廃し、ありのままの現状を把握する必要がある。

手法	例
人に聞く	利用者へのヒアリング、ワークショップ
データを調べる	システムのデータ、統計数値
資料を見る	HP上の制度概要、現行の業務フロー

- 手続のフロー及びステークホルダーのみならず、現状使用されているシステムやその構成（アーキテクチャ）、主要なデータ構造についても棚卸しておくこと。
- 最終的には、整理した業務の全体像に対して、利用者や関連するステークホルダーが抱えている課題をマッピングし、どの領域にどのような課題が存在するのかを可視化し、関係者と合意形成を図ること。

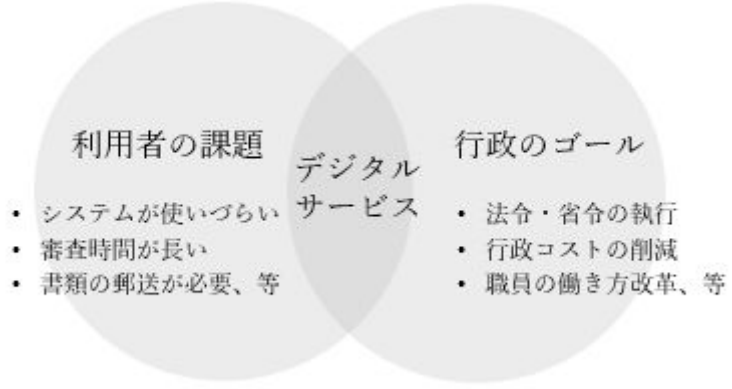
2-2. 行政としてのゴールを定義する

- 現状ベースの問題解決（フォアキャスティグ）では、現状や過去の情報をベースとしたアプローチであるため、従来の延長線上の施策となり、目先の改善に留まってしまう恐れがある。
- 利用者の課題だけでなく、そのサービスを通じて行政機関として何を成し遂げたいのか／成し遂げる必要があるのか、という“行政側のゴール”を設定し、そのゴールに向けてどういう施策が必要なのかを併せて検討する必要がある（バックキャスティング）。
- ゴールの設定に当たっては、行政官等とのディスカッションを通じて、短期（1年以内）、中長期（3年～5年）に目指すべき絵姿を導出する。その際には、手続きという枠に留まらず、実施主体の組織についてもあるべき姿を検討しておくことが望ましい。その際には、課・室⇒部⇒局⇒経産省全体、といった具合に出来るだけ視野を広げて考えること。
- 政府全体のデジタル化との足並みを揃えるため、政府共通方針としての「デジタル・ガバメント実行計画」等も情報ソースとして参照する。

2-3. 課題に対するサービス仮説を立案する

- これまでの手順で把握した、“利用者の抱える課題の解決”と“行政側のゴール実現”の両方を満たす試みが、真に取り組むべきデジタルサービスの要件となる。

デジタルサービスの要件イメージ



- 現状の手続き／課題（As-Is）と目指すべき行政サービス（To-Be）とのギャップを分析し、その根本原因を踏まえた打ち手を検討する
- 課題の原因分析にあたっては、ロジックツリー等を使って課題の原因を掘り下げていき、その根本原因を特定する。行政手続の抱える課題については、以下いずれかの原因分類に帰着する場合が多いが、特定した原因をグループ分けして整理すること。
    - ITシステム
    - 法律（省令／法令）

    - 省内制度
    - 業務プロセス
    - 人
- 特定した根本原因に対して、打ち手の方向性を検討する。それぞれの課題について、ITシステムで解決すべき課題とシステム以外の方法（BPRや制度変更等）で解決すべき課題を切り分け、具体的なアクションに落とし込む。特に制度変更については、行政官の理解が必須のため、関係者と適宜すり合わせを実施すること。
- 解決すべき課題とその打ち手がデジタルサービスの仮説となる。この段階では、あくまで仮説であるため、「サービス戦略」のフェーズを通じた具体化を実施し、サービスデザインのフェーズにおける仮説検証を実施する。

2-4. 費用対効果を踏まえて、解決すべき課題を絞り込む

- 費やせる時間と予算は有限であるため、行政サービスにおいても課題を優先度評価し、よりインパクトの大きい施策から取り組むことが重要である。また、具体的な取組として昇華させるためには、客観的な数値による説明責任が発生する。そうした点から、課題解決の費用対効果を大まかに把握しておくことが必要である。
- 一例として、以下のような数式で費用対効果及び回収期間を計算する。
    - 費用対効果：（課題解決による便益ーコスト）／コスト
    - 回収期間：問題解決による便益／開発期間
- 課題解決による便益については、申請・審査工数の削減やシステム統廃合による運用費の削減といったコスト削減効果が定量的な指標としてあげられる。また、定量化は難しいものの、行政サービスの高度化（データ利活用等）に伴う”付加価値の向上”についても定性的に把握しておくことが望ましい。
- コストについては、この時点では正確な把握は難しい。ただ、正式な事業でなくても、ベンダーによっては”ソリューションや事例の紹介”といった名目でディスカッションやコスト感の算定に付き合ってくれる場合がある。入札時の公平性は担保しつつも、積極的にベンダーにアプローチをし、巻き込んで行くことも一つの方法である。
- また、回収期間も重要な観点の一つである。デジタルサービスはローンチされて初めて価値を生むことになるため、開発期間が長引けば長引くほど、トータルでの便益が少なくなる。課題解決のインパクトと併せて、必要となる開発期間についても考慮することが必要である。
- 複数の課題が存在する場合、費用対効果と回収期間をもとに優先度を評価し、直近に取り組むべき課題を見極める。ただし、行政サービスの特性上、利用者への「公平性」も担保する必要があるため、優先度の検討に際しては、政策的な意義や意図について行政官と十分にすり合わせすること。

2. サービス戦略

1) 本フェーズにおける課題

ビジョン策定においては、主に以下のような課題が存在する。

- 目指すべきサービス像が描けていない（1-1）
- どのようにサービス化を進めるべきか整理されていない（1-2）
- 追いかけるべきKPIを設定できていない（1-3）

1-1. 目指すべきサービス像が描けていない

- どういうサービスを何の目的で作るのか曖昧なままになっており、関係するステークホルダーの腹落ち感がない
- 最終的なサービスのコンセプトが人によってずれており、サービスデザインが円滑に進まない
- 手続きの単なる電子化に終始しており、ワクワク感のある取り組みとして昇華されていかない

1-2. どのようにサービス化を進めるべきか整理されていない

- 対象のデジタルサービスについて、ユーザーの課題を解決するための核となる機能要件が整

理されていない
- 翌年度の活動計画は取りまとめたものの、その後のサービス拡張性などが考慮されていない

1-3. 追いかけるべきKPIを設定できていない

- 担当者の思い付きや興味・関心に沿ってKPIが定義されており、サービスのビジョン実現に向けたKPIになっていない
- KPIの対象がユーザーのみに向いており、システムの運用費や省内の生産性といった観点への配慮が十分でない

2) 本フェーズの進め方

戦略策定は、以下のステップに沿って実施する。
- サービスのビジョンを定義する(2-1)
- サービス化に向けたアプローチを検討する(2-2)
- 改善志向の評価指標を設定する(2-3)

2-1. サービスのビジョンを定義する

- 1.2)2-4にて絞りこんだ課題、及びサービス仮説を踏まえて、目指すべきデジタルサービスの理想像(ビジョン)を設定する。例えば、10年後にはサービスを通じてこういう状態を実現したい、というような最終的に実現したい姿を設定する。
- 具体的には、以下の3点を簡単なストーリーとして説明できるようになること。
    - 何を作ろうとしているのか？
    - 対象とするユーザーは誰か？
    - なぜそのサービスが必要なのか？
- 関係するステークホルダー間でビジョンについて合意しておくことで、後続作業を円滑に進め、モチベーションを維持することにもつながる。

2-2. サービス化に向けたアプローチを考える

- サービスのビジョンを実現するため、効果的なアプローチを検討する必要がある。具体的には、以下のようなポイントを明確にする。
    - サービスの主要な機能
    - 想定されるリスクや制約条件
    - 年単位のロードマップ
    - 行政ゴールへの貢献

- 最終的なデジタルサービスのゴールは単年度で実現することが難しいため、いくつかのフェーズに分解し、複数年度にまたがる活動ロードマップとして整理することが必要である。
- ロードマップを策定したのち、関係者への説明、及び省内の予算要求プロセスに向けたピッチ資料を取りまとめる。以下のような点を盛り込むこと。
    - 現状の課題(As-Is)
    - 目指すべき行政サービス(To-Be)
    - デジタルサービス概要
        - 主要な機能
        - 想定される便益
        - コスト
        - タイムライン

2-3. 改善志向の評価指標を設定する

- サービスのビジョンやアプローチ、および組織としてのゴール等を総合的に加味し、デジタルサービスの改善につながる評価指標(KGI／KPI)を設定する。
- 評価指標の設定に際しては、数値として測定可能なものを選び、その中でも具体的なアクションにつながる指標に絞り込むことが重要である。
- 本稿では、評価指標の設定時に活用できる、大きく4つの”視点”を紹介する。

- 行政の視点
  - 目指す行政サービスの姿にどれだけ近づけているのか
- プロダクトの視点
  - デジタルサービス（プロダクト）がどう使われているのか、また利用者の役にどのくらい立っているのか
- 品質の視点
  - どれくらいのサービス品質を達成できているのか
- 開発の視点
  - システムの開発をどれくらい順調に進められているのかバ

- この4つの視点の中で、それぞれ何を達成すべきか（KGI）を定義し、その達成度合いを測る指標としてのKPIを設定する。KPIについては、複数回ブレイクダウンし、具体的な数値レベルにまで落とし込むこと。

KGI／KPIの設定イメージ（例）

- 必要に応じて、1視点に対して複数の指標を定義することが望ましい

視点	KGI	KPI
行政	審査待ち時間の短縮	・1申請あたりの審査時間 （1担当者あたり1時間）
プロダクト	電子申請の普及	・サイト訪問者数 （1日あたり100人）
品質	安定したサービス提供	・業務停止回数 （1年あたり0回）
開発	計画どおりの機能拡張	・スケジュール通りのデリバリ （1年あたり90%）

- 上記KPIの中でも、Webシステムからデータを取得できるものについては、そのデータを活用してKPIダッシュボードを構築する。

3. 参考情報

1) 共通して活用すべきフレームワーク

- イシューツリー
  https://www.kikakulabo.com/tpl-issue-tree/

- KGI,KPI
  https://www.kaonavi.jp/dictionary/kpi_kgi_ksf/

2) 共通して活用すべきITツール

- ArchiMate
  https://www.sparxsystems.jp/help/15.0/archimate.html

3) まとめるべきドキュメント

1. 全体概要
2. KGI/KPI一覧
3. イシューツリー（課題、制約一覧）

4) 参照すべき政府・経済産業省のルール

- デジタル・ガバメント実行計画
  https://cio.go.jp/digi-gov-actionplan

3.1.2 サービスデザイン

0. サービスデザインとは？

1) 定義

「サービスデザイン」についてWikipediaには以下のように定義づけられているようである。
『サービスの質と、サービス提供者と顧客の間のインタラクションの改善を目的として、人・インフラ・コミュニケーション、そしてサービスを構成する有形の要素をプランニングし、まとめあげる活動である。多くの観察をまとめて、スケッチやサービスプロトタイプなどで、コンセプトやアイデアを視覚的に表現する。』

また、他の企業ではデザインコンサルティング会社のIDEOが提唱した「デザイン思考」と言う概念を用いて具体的なビジネス/サービスに落とし込む手法という意味で使われていたりするが、総じて課題にもなっていない漠然とした疑問の明確化やゼロベースで事業/サービスを興す時などに「デザイン思考」を用いて、ユーザーの視点から解決すべき課題を見いだし、それを解決する為に必要なサービスや機能を考え、試行錯誤しながら作り上げ新しい価値（課題が解決されより満足して利用できる状態）として提供すると言う意味で使っていると思われる。

この「デザイン思考」とは平たく言うと「デザイナーの思考方法を利用してビジネスを考える」と言う意味で、スタンフォード大のDスクールでは下記の様なプロセスを定義している。

1. Empathize：
   ユーザーが何を求め考えているか観察・共感し(≒ユーザー自身になりきり)、ニーズを探る
2. Define：
   調べたことをまとめ、ユーザーの真のニーズを洗い出し、問題を定義する
3. Ideate：
   定義された問題に対して自由に意見を交換し、具体的なアプローチを検討する
4. Prototype：
   試作品を作ってアイデアを具現化し、機能性・効果・実現性について検討する
5. Test：
   ユーザーに実際に使ってもらい、使い勝手、そもそもの問題解決に貢献しているかなどについて様々な点から検証を行い、改善箇所を整理し最終的な解決策を目指す

なお、デザイン思考を具体的なビジネス/サービスに落とし込んだものが「サービスデザイン思考」と呼ばれ、以下のような原則が存在する。

- ユーザー中心（HCD）
- コークリエイティブ（共創）
- シークエンス（インタラクション）
- 見える化（エビデンシング）
- ホリスティック（全体的）

引用：
http://interactiondesigndaysjp.blogspot.com/2013/01/this-is-service-design-thinking.html

なお、サービスデザイン思考の概要については下記YouTubeで簡単に説明がされているので参考にしてほしい。
This is Service Design Thinking – Book Trailer

サービスデザイン思考を実施する際には下記の様なプロセスを採る。
なお、サービスデザイン思考とデザイン思考のどちらも顧客へ「新たな価値」を創造するために用いられるもので、どちらのプロセスもよく似ている。ただし、デザイン思考は顧客の課題を解決するサービスやプロダクトを開発するまでの課程をプロセスに定義しているが、サービスデザイン思考では、それを事業として運営するうえで検討すべき項目についてもプロセスの対象（4. Deliver ）としている。

1. Discover：対象となる顧客の課題を把握するために、ユーザー調査した上でペルソナを定義し、サービスを利用する上での文脈・前提条件を明確にし、カスタマージャーニーマップを用いてサービスを利用する上での現状の顧客の体験・感情を整理(AS-IS)した上で、ペインポイント(解決すべき課題)を見つけられるだけ見つける。
2. Define：個別のペインポイントに着目した解決方法の検討から始め、複数のペインポイントとその解決方法を俯瞰し、全体的な課題を解決できるアイデアを検討・創造する。
3. Develop：創造されたアイデアに基づきあるべき姿のカスタマージャーニーマップを作成し、それに基づいてペルソナの代表に参加してもらい、ロールプレイやプロトタイプ等の手法を用いて評価を行う。
   ※良い評価が得られない場合は、2. Define に戻りアイデアの再検討を行う。
4. Deliver：事業プランを作成するにあたり、ビジネスモデルキャンバスを作成し、関係者・利害関係などを洗い出しビジネスモデルを明確化し、(承認されれば)組織や業務の設計、システム設計などのサービス/事業の構築を行う。
   ※行政の場合、収益構造などのビジネスモデルの検証やビジネスモデルキャンバスにはあまり時間を掛けず、関係者を整理し、業務設計やシステム設計などのタスクを優先的に着手した方がよい。

※Discover～Developまでを別の観点から分割した物がデザイン思考の5つのプロセスと考えて下さい。。

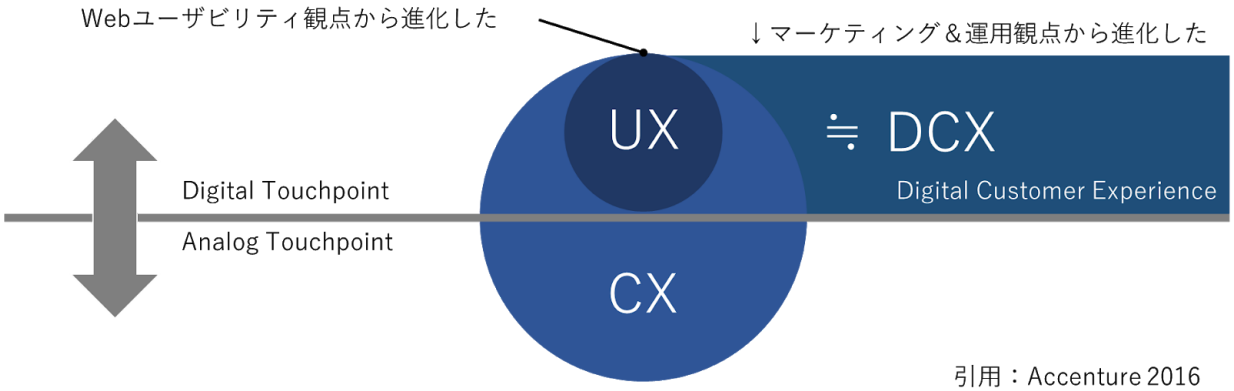
2) UX/CX

UXはUser Experienceの略でユーザー体験と訳され、CXは Customer Experienceの略で顧客体験と訳されている。文章中にどちらか一つだけ使われている時は顧客体験＝UXと理解して良いと思われるが、文脈上『ユーザー体験(UX)』と『顧客体験(CX)』を分けている場合には意味合いの違いについて注意が必要である。

分けて書いてある場合の参考として、それぞれの違いについて、若干乱暴だが、下記の様に整理している。

UX：デジタル上での利用者の体験、Webユーザビリティの観点から進化したもの。※そのためUIとセットでよく語られる。
CX：デジタル・アナログ(コールセンター、郵送物などあらゆる接点)全てを包含する文字通り顧客の体験。

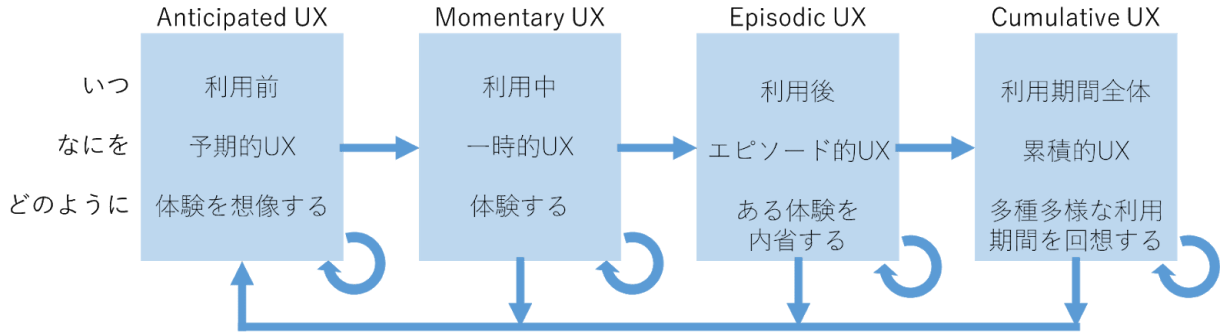
図：UX/CXの関係性



引用：Accenture 2016

また、顧客体験には下記の様な内在プロセスが存在している。しかし、実際に利用者として製品・サービスに触れる際は必ずしも左から右へ一方通行のプロセスではなく常に各プロセス間を行ったり来たり、何度も利用前の体験を行ったりする事、体験を細分化していくと各プロセス内にさらに内在的プロセスを内包する事などを想定してカスタマージャーニーマップやユーザーシナリオを作成する必要がある。

図：顧客体験の内在プロセス



引用：UX白書 http://site.hcdvalue.org/docs

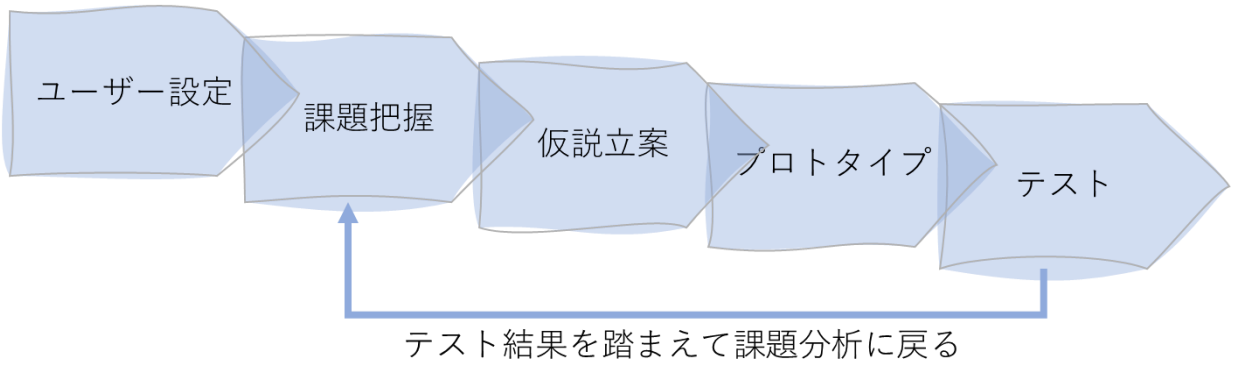
あと、良いUXが必ずしも最適なCXにならないこともあるので、この点についてもUXの改善を検討する場合、注意が必要である。
例：飛行機の場合、着陸後すぐにドアが開かず待たされ、機内から空港の荷物受取場までかなりの距離歩かされる。これは荷物をジェット機からカーゴを取り出し、カーゴから荷物を出し、荷物の安全性/輸入禁止物の持ち込みの有無などの確認を行う時間を稼ぐために行っているものだが、短期的なUXを改善する観点では、着陸後安全が確認されればドアをすぐ開けて、最短ルートで荷物受取場まで案内する事ができる。しかしこの手の改善を行った場合、飛行機から荷物を取り出し利用者の手元まで持っていく時間は変動しないので、荷物受取場で利用者はただ待たされることになり、今まで以上に待つことを強要されたと感じ利用者の航空会社への不満が募ることになる。

人間の時間感覚は状況によって非常に流動的で、ただ待たされると実時間以上に待たされていると認識してしまう。焦っているときなどさらに待たされている感覚が強くなりストレスレベルも上がるのである。
逆に体を動かすなど待つこと以外のタスクを与えて実行している時は時間経過の感覚が短くなり、単純に同じ時間、何もせず待たされる事より待たされていると言うストレス値は低くなる。(心拍数が高くアドレナリンが出ている運動時などは逆に1秒・1分が非常に長く感じたりもする。)
参考：「なぜ時間を長く感じたり、短く感じたりするのですか？」

3) 顧客体験(UX)と顧客体験設計(UXD)とは？

前段でUX=ユーザー/顧客の“体験”について説明したが、ユーザーの“体験”自体はあくまでも個人的なもので、そのユーザーが今置かれている状況、過去に見聞・体験したこと、このサービスを触れる前に期待していた事、サービスを見たときの第一印象(認知バイアス)によって変動する。
サービスを提供する側として「サービスを通じてどう言った“体験”をしてもらうか？」について考える必要があるが、顧客体験設計(UXD)とは、特定のユーザーではなく対象セグメントに存在する不特定多数のユーザーが同種の“体験”を量産・再生産されるサービスと仕組みを仮説として設計し、プロトタイプを通じて設計した仮説を検証し改善するというサイクルを継続的に回し続ける必要がある。サービス上で想定している顧客群が何度でも同じような体験を享受出来る様に下記のステップを踏んで設計して行く。

図：顧客体験設計のプロセス



4) 注意すべき事

<u>・サービス提供者はユーザーではない</u>

UXの設計者を含むサービス提供側の人間は、サービスの仕様・その他前提条件などの理解、Webサービスなどへの ITリテラシなどの観点で、実際の利用者と異なる為、サービス提供者側の人間がユーザーの立場に立ったとしても、本当の意味で利用者の意見とは一致しているかどうか分からない(実体験として、一致していないことの方が多い)。
もし、サービス提供側の人間が機能要件や改善箇所について検討を行う場合は、解釈の余地が入らない“定量的な情報”や事前に設定したペルソナの行動・心理を鑑みた上で検討を行う事で健全な議論と課題の整理が行うことができる。
検討の場において、関係者の個人的な考えや、ユーザーの意見を代弁するような発言を行うことがあるが、これは実際のユーザーの意見ではなく、適切な議論を阻害する要因になりうるので、そういった発言をされる方に対しては設定したペルソナに基づいて自身の発想や意見を関係者と摺り合わせる様に促すこと。

<u>・仮説と検証</u>

どれだけ調査を行い、ペルソナを作り、ユーザーシナリオを作ったとしてもそれは全て「仮説」の範疇を超えていない。仮説は常に検証して行く必要があり、そのために必要なドキュメントやKGI/KBR/KPIなどの評価指標の設定や定性・定量の両面から検証(計測)可能な状態を構築する必要がある。これらのドキュメントがないとリリース以後の仮説検証やサービス品質・体験の継続的な改善に繋がらないので、アジャイルと言う名目でドキュメントを作成・適宜更新しないベンダーには厳しく作成する事を求めること。
そして、検証した結果当初の仮説が間違えていたと分かった場合は、間違えたことが問題ではなく、そこで思考が停止して、仮説検証の手が止まらないように、間違えた人を責めず前向きに

仮説の再設定からやり直すようにすること。

・顧客体験設計および改善

前段に書いたとおり短期的なUX改善が必ずしも最適なCXを提供できるとは限らなん。その為ユーザーが直面している課題/ストレスについてユーザーが体験するジャーニー全体において、ユーザー自身が乗り越える必要がある課題なのか十二分に考える必要がある。

また、UXDは顧客体験の設計を通してステークホルダーなど全ての関係者の共通理解・合意を形成するプロセスとして活用するべきものなので、UXDを担当する人に全て任せておけば良いという事ではない。(専門家に任せれば全てOKという思考停止はNGである)
UXDを行う人はユーザー体験デザインの専門家であったとしても、そのサービスについて熟知している訳でも、ビジネスゴールや利用者との関係性を熟知しているわけでもない。そういった点を加味して関係者全員で顧客体験を設計出来る様に体制(アサイン、リソースの確保)を組んでいくこと。

・課題が分かっている場合

関係者の共通認識となっている課題がすでに存在している場合は、わざわざゼロからペルソナを設定して、課題を抽出する等の一連のサイクルを最初から回す事に時間を割くよりも、顕在化している課題に対する改善策、改善効果、改善が全体に及ぼす影響などの仮説を立てて素早く改善のサイクルを回した方が、実際のユーザーの行動や反応を把握することができ、その反応に基づいた改善やより精緻なペルソナの設計などに活かすことができる。
想定した課題がまだ漠然としている、課題について関係者の合意が得られていない等の場合は、UXDやユーザビリティテストの手法を通じて共通認識を醸成する。

・デザイン思考への過信

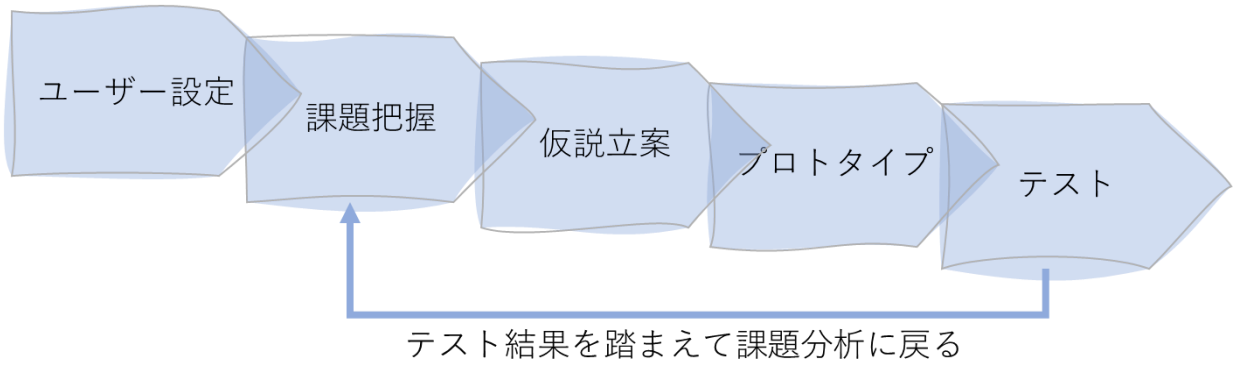
デザイン思考という手法を採れば必ず最良の改善策が発見できると言う訳ではなく、唯一無二のサービスが出来上がる訳でもない。
デザイン思考は確かに一般的なデザイナーの思考の順番に基づいた手法になっているが、本来デザイナーは検討の過程で自身のアイデアに対して、第三者的な視点で批評しそれを改善するという内部サイクルを何重にも回した上で、アイデアをプロトタイプに進めていく。デザイン思考には批評というステップが抜けているため、正しい手順で手法を実施したとしても出来上がったアイデアやシステムが必ずしも最良な物ではない事を前提としてアイデアやシステムを常に検証出来る状況を作る事を心がけることが重要である。
参考:

- AXIS:「デザインシンキングなんて糞食らえ」ペンタグラムのナターシャ・ジェンが投げかける疑問
- なぜデザイン思考はゴミみたいなアイデアを量産してしまうのか
- 「デザイン思考なんて糞食らえ」ナターシャ・ジェンが投げかけた問題について考えてみる。

1. 進め方

UXDの考え方を通じてサービスをデザインするためには以下のプロセスを経て進めていく必要がある。(再掲)
また、各プロセスを進めていく上で、できる限り全てのステークホルダーを巻き込んで実施していく必要がある。
もし、経営者や管理職などの時間の確保が難しい方が居る場合は最低限ユーザー設定～仮説立案までは参加出来るように調整すること。
どうしても、参加出来ない人には適宜情報を共有し進捗と関係者の現状認識を理解して貰うよう努める。それも難しい人は本プロジェクトの関係者としてはふさわしくないので外れて貰う方が後々トラブルにならなくて良いと思われる。

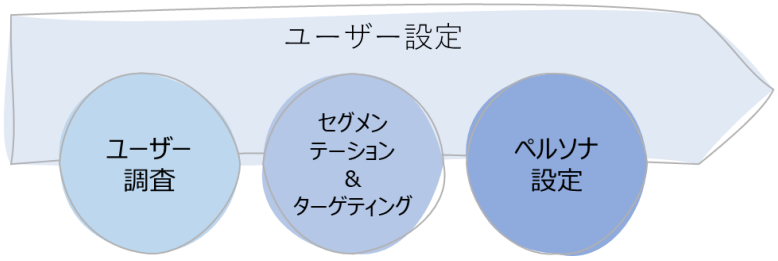
図:顧客体験設計のプロセス



- **ユーザー(ペルソナ)の設定**

サービスの対象とするユーザーを定義する際には下記のようなプロセスを通じて行う。

図:ユーザー設定のプロセス



- **ユーザー調査**

どう言った人物がユーザーになり得るのか、想定ユーザーのペルソナを定義する為には、まずインタビューや行動分析などユーザーに関する下記の様な調査を実施する必要がある。

A) ユーザーインタビュー
B) エスノグラフィーなどの行動様式調査
C) 行動分析
D) エクストリームユーザー調査

A) ユーザーインタビュー

ユーザーインタビューは、対象のサービス及び事業領域について実際の利用者や想定顧客層に対して実施する。
本インタビューは自身の仮説などを検証するために行うものではなく、あくまでもペルソナを定義するためにユーザーの思考や行動からユーザーのストーリーやインサイトを抽出する目的で行う。その為、インタビューを実施する際は「はい/いいえ」や設問中に提示された選択肢の中から回答する様なクローズ型の質問ではなく必ずオープンエンド型の質問を投げかけ、ユーザー自身の考え、行動における指針などをできるだけユーザー自身の口から自由に語って貰う様に聞き役に徹する意識でインタビューを実施すること。
詳しくは下記サイトを参照ください。
参考:U-Site – ユーザーインタビュー:実施の理由・方法・タイミング

また、質問を設計にするにあたり、Excelなどで質問したい項目を書き出していく事が多いかと思われるが、KJ法を用いると、複数人で項目を検討する際に整理がしやすくて効果的である。
参考:OpenCu:樽本徹也 意外と知らない“KJ法”のオモテとウラとは。第2回ワークショップレポート

インタビューを行う際に注意して欲しい事として、ユーザーインタビューの結果を踏まえてペルソ

ナやシナリオが作られ、課題やニーズが抽出されるため、インタビューに失敗すると、プロジェクト全体の方向性を誤る可能性が高いため、以下の点に注意すること。

1. 誘導型の設問は避ける
   質問者の言葉の使い方や使う単語・タイミングなどで、インタビュー被験者の前提認識やサービスへの期待値を変えられるため、設問及びその前後の会話などには細心の注意を払うこと。
   例として、「将来的～」「改良された～」「最新の～」等の言葉を使った場合、被験者は無意識に良い印象を抱く可能性が高くなる。同時に「過去の～」「古い～」等の言葉は無意識にネガティブな印象を抱く可能性が高くなる。
   また、被験者が質問者の発言にかなり影響を受ける事を考慮に入れ、内容に共感するなどの反応を行う際は、自身の考えなどを組み合わせて発言する事は行わず、被験者に感情及び感想を言わせるようにする。
2. バイアスを外す
   人間は普段から無意識のうちにあらゆる事柄に対して自分の価値観や体験などを元に主観的に理解・判断する。そういったバイアスは人それぞれで、質問者にとって“誰でも知っている当たり前/常識“のことが、被験者にとっては”未知のこと/非常識“な事柄の場合がある。設問を設計する際には、質問者自身に知らないことがあるできるだけ質問者自身のバイアスを外すよう、設問の前提を書き出し、そこに自身のバイアスが含まれていないか検証すること。
3. 複数の質問を同時に行わない
   被験者が回答した内容に対して複数の疑問が湧いた場合、『〇〇について**困ったこと**や**不満だと思う事**はありますか？』といった感じで質問すると、被験者は混乱しどちらの質問についても不完全な回答を行う可能性が高くなる。その為、まとめて質問するのではなく、順序立てて1つずつ簡潔な質問を行うようにすること。
4. 適切なフォローアップをおこなう
   被験者は質問者が思っている以上に緊張していたり、質問者を警戒していたりする。その為、インタビューの序盤に興味や共感を示すようなフォローアップの質問を行う事で、被験者も自分の話に興味を持ってくれていると感じられ、被験者の緊張や警戒心を解いて、より多くの情報を提供してくれることになる。
   会話例：
   A「〇〇というサービスを、無形固定資産の管理の為に使っています」
   B「〇〇の**どのようなところ**が役立っていますか？」
5. インタビューでは用意した質問を用意した順番に聞けば終わりではない
   ユーザーインタビューはユーザーの発言からペルソナ設定やサービスの課題やニーズの発掘を行う事を目的にしているため、当初設計した設問に全て答えて貰う事を優先しがちだが、全て聞くことが重要ではない。実際にインタビューを行うと被験者が答えてくれた内容に対してどんどん新たな疑問が湧いてくると考えられる。その為、用意した設問の内容はあくまでも質問が脱線しすぎないようにする為の目印として考え、被験者が話してくれた内容に疑問が湧けば、聞きたいことをその都度追加して聞けるだけ聞き、より深い情報を得るようにする。
   なお、質問者が設定した設問には質問者の“当たり前“というバイアスが含まれており、質問者が「知りたいこと」は本当の意味で「知りたいことの全て」ではない。本当に知りたいことは、被験者と質問と回答を繰り返して積み上がった結果から見いだされるもので、最初から想定できるものではない。
   単に質問項目に答えて貰う為だけならば、質問シートを渡して埋めて貰うだけの方がお互いにとって都合が良いが、当然被験者の深掘りなどができないので、質問シートに埋められた回答だけでインタビューを実施しない等の手抜きがないようにする。
6. 被験者に必要なものを聞かない
   被験者自身、自分に必要なものを分かっている人はほとんどいない（※エクストリームユーザーを除く）、必要なものを答えたとしても即物的で根本的な課題に根ざした物でない可能性が高いので、安易に「課題は何か？」「〇〇に追加すべき機能は何か？」といった質問は行わず、被験者がサービスを使って困った体験など被験者自身の体験に焦点を当てた質

問を何度か行いその回答を通して被験者の課題やニーズを抽出するようにする。
参考：ユーザーは欲しいものを知らないから行動から見つけよう

7. 「なぜ」と言う言葉は使わない
「なぜ」という言葉はより深く情報を得るために利用する言葉の様に思えますが、実際に使うと被験者は自身の意見を否定されている様に感じたり、問い詰めている様に受け取られたりするため、被験者は質問者に対して不満や反発を覚える。回答の意図を知りたい場合や回答内容を深掘りしたい場合などは、「なぜ」と言う言葉を使わず『教えて下さい』と言い換えて、被験者の心理的なストレスを抑えてより自分の意見を言いやすい状況を作る。
8. 適切な間・表情・感情で反応する
インタビューは被験者との対話によって被験者の意見を引き出すので、PCに視線を向けたまま話した内容に対して一方的に質問するだけでは、被験者の警戒心や緊張をほぐすことができず良いインタビューにならない。また、被験者と質問者の間で意図せず上下関係が生まれてしまうことも多々あり、被験者が緊張で萎縮することや、警戒がとけないことがある。前段で警戒心や緊張をほぐすために興味や共感や示す質問をする事を書きましたが、それと同じように被験者の目や顔を見て質問し、被験者が自分の言葉で語るまで沈黙して待ち、被験者が話した内容及びその時の被験者の表情に合わせて適切な表情・感情で対応することで、相手に安心感や上下関係の無いフラット関係であることを認識してもらう必要がある。また「ミラーリング」という心理テクニックを用いて被験者に親近感や安心感を抱かせる手段も有効である。
あと、被験者はインタビュー中、常に不安な状態におかれているので、質問のあとに、被験者が話した内容の中でどこか自分にとって新鮮で、どんな価値があったのかを分かりやすい言葉で被験者に伝える。（安心する）
9. 見送って別れるまでがインタビューである
インタビューが終わってICレコーダーを切ったあと、被験者をお見送りできるところまで見送ること。見送る際にインタビューを振り返りつつ、話を聞いて下さい。被験者は被験者の役割が終わった事で、緊張が解け「実は～」と本音を話し、先ほどまでの回答内容と異なる内容の話をすることが“良く”ある。こちらの方が重要な情報なので、雑談と思われる内容だとしてもちゃんと意識して記録に残すようにする。

B) エスノグラフィー

エスノグラフィー自体は民俗学や文化人類学等で使われている研究手法である。
実際に、特定の消費者の生活様式を理解し行動を観察し、疑問に思った行動についてその場でインタビューして理解することで、観察者の先入観が入らずニーズを掘り起こす事ができるので、観察者の思いもよらないニーズや気づきを得る事ができる。

実際にエスノグラフィーを実施する場合、対象者に普段通り業務を行って貰い、観察者が業務内の行動を観察し、行動に対してその都度「なぜそうしているのか？」を聞くことで対象者の本音を聞き出し、潜在的な問題を浮かび上がらせることができる。
※ユーザーインタビューとは逆で、「なぜ」を聞きまくること

観察を行う際に、観察者自身が対象者の業務に参加し共に行動しながら調査すること（参与観察）で、より具体的な問題の把握や改善案を発見する事ができる。
参考：U-Site：エスノグラフィー調査

C) 行動分析

行動分析は物品の購入など行動結果を起点として時間軸を遡りながらユーザーを取り巻く環境から何を見て何を感じ・考え、どのような行動をとっているのかの因果関係を明確にし、それぞれ次の行動に繋がる要因を調査・分析する。
実際のユーザーの行動を観察することで、インタビューとは異なりユーザーを取り巻く実環境やユーザーが無意識に行っている行動や言動を把握・共感する事ができ、サービスにおけるユー

ザーシナリオ、ペルソナ設定の際の人格・判断基準・行動傾向、サービスにおけるユーザーの課題などの要素を発見する事ができる。
また、行動分析を行った上でインタビューを実施すると、より詳細にユーザーの認識や課題を把握することができる。

観察した内容を整理する方法はいくつかあると思うが、観察内容を元にユーザーが最終的な目的に到達する為の行動をステップ毎に書き出し、ステップ毎にユーザーを取り巻く環境（見た物、触れたもの、認識したもの）とユーザーの感情・思考を整理する。もしユーザーが無意識の内に行っている行動があれば、発生したステップに書き記すが、無意識下での行動と次のステップの行動に因果関係があれば関係性が分かる様にステップの前提条件として記述する。

また、整理した内容を元に行動分析学の観点から下記の行動随伴性と呼ばれるユーザーが行った行動と直前の環境の変化の関係を整理しその要因が行動頻度を強化するものか、弱化するものかを分析する。

4つの行動随伴性：
- 正の強化：好子（自発頻度を高める刺激）出現による強化
- 負の弱化：好子消滅による弱化
- 正の弱化：嫌子（自発頻度を低める刺激）出現による弱化
- 負の強化：嫌子消滅による強化

参考：
行動分析学－Wikipedia

既存サービスのWebサイト上での行動分析では、Google AnalyticsやAdobe Analyticsなどのアクセス解析ツールや、ヒートマップツール、マウスやブラウザの動きを計測するツールと実際の利用者の顧客データ、購買データ、同一IDを使った他サービス利用履歴等のデータを活用してセグメンテーションやペルソナの設定、購入などコンバージョンとして設定したポイントへたどり着くまでに存在する阻害要因の洗い出しを行う。

D) エクストリームユーザー調査

エクストリームユーザーとは、特定ユーザーセグメントやターゲット層の範囲内の中で大多数を占める行動とは異なる極端な行動や性質を持つ人を指す。※ヘビーユーザーとは異なる。
なお、エクストリームユーザーには大まかに3種類存在しており、より適切にサービスを設計する上で、全ての層を調査する必要がある。

1. プロフェッショナルなユーザー
   その行動にこだわりを持ち量/頻度/継続期間において、時間とコストを掛けているユーザー。先行的に行っている事や知識を、市場（競合）より先に把握するために調査を行う。
2. ペルソナに近いユーザー
   あまりにペルソナの想定通り過ぎるように見える人物。これはエクストリーム調査と言うより、ペルソナに関する仮説検証を目的として調査を行う。
3. ネガティブで逃げ腰なユーザー
   プロフェッショナルなユーザーとは反対にターゲットの範囲内にもかかわらずに行動の量/頻度/継続期間において、極端に低い人物である。この物やサービスをほとんど利用していない人達を調査する事で観察者が予想もしなかった代替的問題解決手法を発見することがある。

調査を実施する上で、注意すべき事は下記の通りである。

- 相性が良いのは新規サービス開発のみ

既存ニーズを満たす競合サービスがすでに存在している状況で新しくサービスを作る場合、何かしら革新的な体験や価値がなければ、ユーザーの興味を引いたり、利用者を増やしたりすることは難しい。革新的なサービスを提供するためにはエクストリームユーザーが先行的に行っている行動の本質を捉え、より一般に普及しやすい形に翻訳する必要がある。

- エクストリームユーザーがサービスのメインペルソナではない
  エクストリームユーザー調査を行うとエクストリームユーザーに対してサービスを提供したくなると思うが、そもそもエクストリームユーザーは少数派なので、彼ら向けのサービスを提供したとしても事業は成功しない。サービスのメインターゲットであるユーザーに対しての新しい価値の提供のための手段と捉え、エクストリームユーザーの先行的な行動によるより良い体験をメインターゲットのユーザーでも得られるように一般化した機能を開発・提供する。※エクストリームユーザー向けのサービスはそれまでエクストリームとしていたユーザーがメインユーザーとなり『エクストリーム』ではなくなるので、再度エクストリームユーザーを探すところからやり直すことになる。。

- **セグメンテーション/ターゲティング**

  ユーザー調査の結果に基づき、サービスを利用する想定ユーザーを決める為にセグメンテーション（市場細分化）及びターゲティング（切り分けられた有意な集団の選択）を行う。

  ユーザー調査を通じて、漠然とした想定ユーザー群の構成条件などがいくつか浮かんでいると思うが、それらを元に様々な軸（セグメント）を設定し市場を有意な集団に切り分け、切り分けられた集団の中からサービスの対象顧客としたい集団を選択する。

  セグメントの例：
  - 地理、年収
  - デモグラフィック（人口統計）
  - サイコグラフィック（心理傾向）
  - 購買行動
  - 購買心理
  - 利用/来訪歴

  など

  なお、セグメントを実施する場合の注意点として、年収だけ、地域だけで切り取らないようにすること。地域によって基本的な支出額が異なり、首都圏で年収500万と地方で年収500万では生活様式が異なる。もしお金の軸で切り分けたい場合は、可処分所得やお小遣い額などでセグメント条件を設定する。
  様々なセグメント条件で切り分けていくと、切り取られた集団がそもそも存在しない、他の集団に比べて以上に多いなど、いくつか問題が発生するので、その場合は適宜セグメント条件を調整し、様々な視点で矛盾や破綻がないか確認すること。
  ※多少の集団規模の揺れは有って良いのですが、他の集団の2倍以上の集団が存在する場合は、他のセグメントで切り分けられないか検討が必要である。
  また、事業戦略上（と言うか経営陣の希望として；）、優先して獲得すべき理想的なユーザーの集団のイメージが存在している場合、そもそもその集団が存在するのか？リーチ＆獲得可能か？規模として適当か？獲得した場合の二次的効果は？等の観点から検証を行い、各集団の優先順位を設定する。

- **ペルソナ/プラグマティック・ペルソナの作成**

  対象ユーザー像について誰でも簡単に理解を深め、イメージできるように具体的にする事で、認識の共通化、顧客視点での品質改善、マーケティング活動の効率化を実現するために選択した集団のセグメント条件およびユーザーインタビュー結果に基づき架空のユーザー像（ペルソナ）を作成する。
  もし、戦略的な意図でペルソナの中で優先順位を付ける場合は、

基本的に、選択した集団毎に最低1人設定するが、もし予算・時間的な制約でユーザーインタビューなどのユーザー調査が実施できなかった場合は関係者の想定しているユーザーイメージを摺り合わせる目的でプラグマティック・ペルソナを作成しプロジェクトの中で適宜ブラッシュアップする。

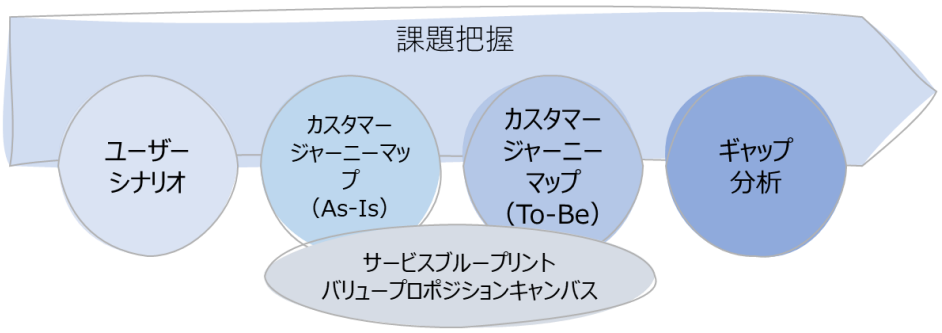
参考：

- Web担当者Forum：ペルソナとは
- PLAN-B：ペルソナ設定とは？具体的な手法や注意点も説明！マーケティングには必須！
- tree：良質なペルソナ分析と作り方・手法を公開
- ペパボテックブログ：プラグマティック・ペルソナを作ってユーザーを理解する
- Qiita：もう、使えないペルソナは捨てよう! プラグマティック・ペルソナのつくりかた
-

- **課題把握**

課題を把握するには下記の様なプロセスで現状のユーザーの行動とあるべき姿とのギャップを洗い出す必要が有る。また、ギャップは抽象化せず、できるだけ詳細な情報を得られるように深掘りすること。

図：課題把握のプロセス



a.　ユーザーシナリオ作成

ユーザーシナリオは以下のような流れで作成を進める。

1　ベルソナ毎のサービス利用開始前～申込～退会までの大きなシナリオ作成
2　ステップ分解と各ステップの目標とタスクの設定
3　ステップ毎のユーザーシナリオ作成

シナリオは各ペルソナが行う行動毎に定義する必要が有るが、いきなり個別のユーザーシナリオから書き始めると、網羅性が足りず不十分なシナリオが出来上がることがある。それを避けるためには、まず、対象のペルソナが当該サービスを利用する前から利用を終了（退会）するまでの大きなストーリーを書く。
大きなストーリーを書いた上で、行動分析結果などを元に大きなストーリーをステップに分割しそれぞれにおいて発生する可能性の高いユースケースを定義し、ユースケース毎にペルソナが達成したい目的と達成するために行うべきタスクを定義する。
定義された内容を元にペルソナが実際に行動する体でシナリオとして書き出していく事で網羅的にシナリオを作成する事ができる。
また、シナリオを作成する際には以下のような情報を定義する必要がある。

・　このシナリオのゴール
・　ペルソナ名
・　ペルソナのシナリオ上での目的
・　流入経路
・　シナリオ番号・名称
・　ペルソナが辿るステップ

また、よりユーザーの行動を理解する為に、実際の行動だけではなく行動の前提条件となる以下の情報についても記載する。

・　ペルソナの環境

職場、自宅、レンタルオフィス、出張先のホテルなどペルソナが対象のサービスに触れている場所を特定する。

・　ペルソナの精神状態・思考

ペルソナが遭遇している場面を想像して、ペルソナの心に何が浮かぶかを想定する。

・　動機

ペルソナがそのサービスを利用するに至った理由を知ることで、サービスを利用するための目的や、サービスを利用することになったきっかけを想定する。※動機がズレたら最終的な結果もズレる。

・　影響を与える外的要因

ペルソナの周辺で起こる工事などの騒音や利用者の長蛇の列、ネットの回線速度やサービス利用時間、利用デバイスなどペルソナの行動や思考にポジティブ/ネガティブな影響を与える外的要因を整理する。

なお、個別のペルソナのシナリオを書き出すにあたり、他のペルソナで作成したシナリオと類似している等の理由でシナリオ作成を省略したいと思われるかも知れないが、類似しているだけで詳細において異なる可能性が高いので、全て書き出し細部を確認した上で、マージするかどうかを検討する。

参考：Goodpatch Blog：テストを考えるときはユーザーシナリオを作ってみよう

b.　カスタマージャーニーマップ作成(As Is)

現状を把握する目的で、ユーザーシナリオに基づいてAs-Isのカスタマージャーニーマップ(以下、CJM)を以下のようなステップで作成する。

1　シナリオに基づいたユーザーのアクション設定
2　アクション毎のタッチポイント定義

3　タッチポイント毎にユーザーの目標(達成したい事)を設定
4　タッチポイント毎にユーザーが思っていること(喜び、不平不満など)
5　感情/ストレスレベルのグラフ化

基本的に前段のユーザー設定の段階で行ったユーザーインタビューや行動分析などの結果と設定されたユーザーシナリオに基づいて作成される。
記載する内容はできるだけ抽象化せず、具体的・詳細に記載する事を意識する。
また、検索エンジンで「カスタマージャーニーマップ」と言うキーワードで検索すると、様々なものが並ぶが、概ねAISASやAIDMAと言ったフレームワークに基づく物になっている。これらはCJMに似て非なる“何か“であって、本当に意味のあるCJMではない。これらを紹介しているサイトに従って書いても改善ポイントもギャップも抽出できず、意味のあるKPI設計もできない。(経営陣を分かったつもりにさせると言う観点では有益だが、事業としては作るだけ時間の無駄である)
CJMはペルソナの体験をジャーニーとして表記する物だが、ユーザーの体験はAISASなどで定義しているように都合良く次のステージに移るものではなく、実際には前段の「顧客体験の内在プロセス」でも言及したとおり、各ステージを行ったり戻ったりしながら進んでいくため、内在プロセスが有ることを前提にしてCJMを作る必要がある。
参考:Web広告研究会:「そのカスタマージャーニーでいいんですか？ UX実践者が教える正しい作り方と使い方」 016年4月8日開催　第10回東北セミナーレポート　第2部 イベント報告

なお、As-IsのCJMはそもそもユーザーシナリオや行動分析の結果があるので、AISASなどのフレームワークにはめてCJMを書かないように気をつけること。

最後に、ユーザーの行動や認知やユーザーの経験の蓄積、環境の変化、流行、時代の変遷などによって変化するため、サービス検討時に設定したAs-IsのCJMはサービスリリース後の運用において実際のユーザーの行動に合わせて改修して行く必要がある。※調査結果に基づいたとしても、その当時のものであって、現在も同一か？と言う疑問を持つことが重要である。

c.　カスタマージャーニーマップ作成(To Be≒夢)
現状のカスタマージャーニーマップが書けたら、今度は本来あるべき姿(そうあって欲しいと言う願望や夢)のCJMを書いていく。
有るべき姿のCJMを書く際には、記載粒度をAs-Isと合わせる。また、検討に際して現状のサービスを構成する上での前提条件などの制約をできるだけ排除して考える必要があるが、新しいサービスを構築したとしても共通で存在するユーザーIDとか、本サービスと連動する他サービスなどの存在・仕様は、予め折り込んで作成する。

おそらく、障害もなくストレスもない、使っていて心地良いと言う、非常にハッピーなCJMが生まれると思うが、技術的な実現可能性を考慮して必要であれば、やり直すこと。

例えば、観光旅行におけるCJMを書いたときにAs-Isでは「飛行機で長時間移動する」と記載していたところをTo-Beでは「ワープして目的地到着」と
していた場合は、ワープすると言う手段を選択した背景課題や仮説を考慮して、そこを解決したCJMを書き直すとよい。

あと、これらは仮説に基づいたもので、あるべき姿はAs-Isと同様に時代の変遷や事業構造・競合の存在など内的・外的要因によって変化して行くため、以前作成したTo-BeのCJMが間違っている可能性がある事に注意する。
To-BeのCJMはAs-IsのCJMと同様にサービスリリース以後に実施した調査やアクセス解析結果によって、定期的に再定義すること。

d.　ギャップ分析
現状のサービスの課題把握する為にAs-IsとTo-BeのCJMを比較するとユーザーの感情(ストレス)や得られる情報などいくつかのギャップを発見することができる。
※As-IsとTo-Beではタッチポイントで得られている情報が異なったり、As-Isでは感情のグラフ

が降下しているが、To-Beでは降下せず高い位置を維持していたりする部分が、ギャップになる。

もし、ギャップを発見した場合は、後段の課題の解決(仮説の立案)を容易にするために、ギャップの構造とその発生原因を具体化・詳細化するべく「どうやったらTo-Beの状況に到達できるのか？」「何が原因で到達できていないのか？」という問いかけを繰り返し、その問いかけを通じてギャップの詳細や裏に隠れている制約や課題などを抽出・整理する。

なお、CJMのAs-IsとTo-Beの記載粒度が異なるとギャップの構造が不明瞭となり、具体的な解決策を検討する妨げになる。そのため、As-IsとTo-BeのCJMで記載粒度が異なっている場合は、前のステップに戻り、記載粒度をより詳細なほうに合わせて修正していく。
※As-IsがTo-Beより細かすぎる場合はさほど問題は無いが、逆の場合は正しい課題の抽出が難しくなる。

課題把握が一番重要なステップなので、矛盾や不明点、ギャップ自体が不明確な場合などは、前段のAs-Is、To-BeのCJMを精査し必要があれば作成し直すこと。
参考:
- 理想と現状を比較して問題を可視化するフレームワーク「As is / To be」【問題発見】
- ギャップ分析 + 問題発見の4P　～現状とあるべき姿のギャップを分析

e.　サービスブループリント

サービスブループリントは、カスタマージャーニーに存在するタッチポイントに直接関係している人、モノ(物理的・デジタルな証拠)、プロセスといったサービスにおける各要素間の関係性を視覚的に表現したダイアグラムである。※決まったフォーマットがないので、適宜設定すること。

そのため、作成にあたりカスタマージャーニーマップとして定義した、ジャーニーに基づいて各タッチポイントで要素の整理分解を行う。
整理する要素は以下の4つである。

- 顧客のアクション
  ユーザーの目的を達成するために、サービス上でユーザーが行う行動や選択、やり取りなどを指す。これらはCJMから抽出して記載する。
- フロントステージのアクション
  タッチポイントで発生するユーザーが直接見ることができる反応/アクションである。アクションにはユーザー(人)対サービス提供元の従業員(人)、ユーザー(人)対テクノロジー (コンピューター)がある。
  ※タッチポイント全てフロントステージのアクションが常に起こるとは限らないことに留意すること。
- バックステージのアクション
  フロントステージでのアクションをサポートするために舞台裏で発生するステップや反応/アクションのことである。
  この反応については直接ユーザーが見ることはできない。
- サポートプロセス

フロントステージやバックステージで活動する従業員をサポートするアクション及び組織内のステップのことである。

また、サービスブループリントでは、各要素を分ける下記の様な境界線が存在している。

- インタラクションの境界線(line of interaction)
  直接的なやり取りが行われている、ユーザーと提供側の組織との間に設定される
- 可視境界線(line of visibility)
  ユーザーに見えるサービス全てとユーザーが直接見ることができないバックステージの間に設定される

- 組織内のインタラクションの境界線（line of internal interaction）
  ユーザー担当窓口の従業員とユーザーとのやり取りを直接的にサポートしない従業員との間に設定される

これらを用いて整理する事により、ユーザーのタッチポイントにおける行動や体験の裏にあるサービス提供元従業員の活動やそれを支援する従業員などの活動を全体的に捕らえられるようになり、業務の重複や不要な依存関係などサービスを提供している組織上の課題を発見しやすくなる。

参考：U-Site：サービスブループリント：定義

f. バリュープロポジションキャンバス

バリュープロポジションキャンバスとは、CJMなどのユーザーの体験にフォーカスしたモノとは異なり、現在提供しているサービスと顧客のニーズのズレを発見し解消するために使用するフレームワークである。

ユーザーは提供されているサービスが自身にとって有益(価値がある)だと認識したモノだけ購入・利用するが、実世界においてサービス提供元が想定しているユーザーニーズやそれに基づいた提供価値とは、別の価値を実際のユーザーが見出していることは良くある。
このズレを放置しておくと改善すべき課題を見誤り、ユーザーが求めていない改善（ユーザーからは改悪と認識）を行ってしまい、ユーザーを失ってしまう。このズレを認識し適切に改善する事でより多くのユーザーに受け入れられるサービスに変わる事ができる。

バリュープロポジションキャンバスを作成する場合は下記の様な手順をたどる。

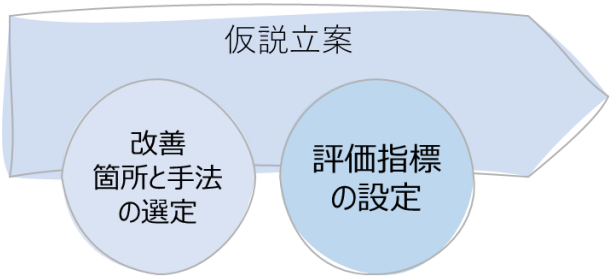
1. 対象ユーザーを設定する(選択した有意な集団に基づいたペルソナ)
2. 対象ユーザーにサービス提供元が提供する価値
3. 対象ユーザーが解決したい課題(Job)
4. 対象ユーザーが課題を解決することによって得られるもの
5. 対象ユーザーが課題を解決するにあたっての悩み
6. サービス提供元が提供している製品やサービス
7. 提供サービスによって顧客にもたらされるもの
8. 提供サービスにおいて顧客の悩みを取り除くもの

参考：バリュープロポジションキャンバスの重要性と作り方を徹底解説します

- **仮説立案(課題解決＆To-Beの実現に向けた)**

前段のギャップ分析などによってAs-IsとTo-Beのギャップとそこに存在する課題の把握、ギャップを埋める手段などについて検討していたと思う。
ここでは前段で発見した課題に対する改善策を仮説として設定する。
※課題も含めてすべて仮説といえるが、ここでは課題解決を仮説と呼ぶ

図：仮説立案のプロセス



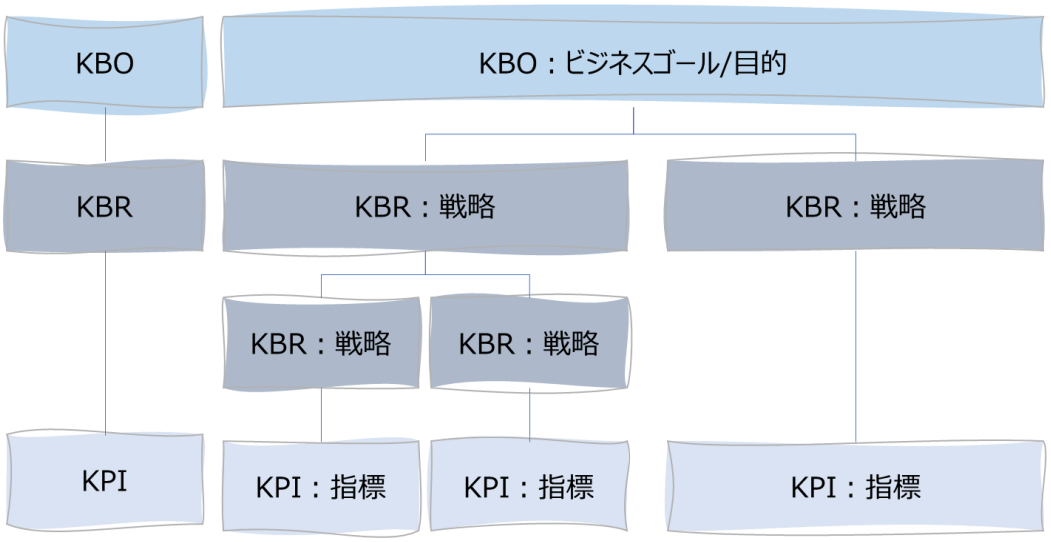
a. 仮説として改善箇所の選定および改善方法の検討

改善箇所に既存サービス(Web)がある場合は、前段で検討した内容に加えてアクセス解析ツールを活用し課題解決に繋がる改善箇所(仮説)を探す。既存サービスがない場合は、前段までの検討結果を基に課題を解消する改善箇所と改善方法について仮説を立てる。
ユーザーの一連のジャーニーの中に複数の課題が存在する場合、課題毎の改善策を俯瞰し、全体的視点で課題をより少ない改善で解決できるように改善策を最適化する。

b. 評価指標設定(KBO→KBR→KPI)
仮説を立てる場合、仮説が正または誤であることを検証できる状態にする必要がある。
そのため、「改善策を実施することでどう言った変化を生むのか？」「それはどの程度変化すれば成功と言えるのか？」などの質問を通じて、定量的で計測可能な評価指標を設定する。

また、単に仮説や各改善施策に基づいて評価指標(KPI)を設定する場合、サービス全体を通して見たときに適切な指標を設定できていないことや、指標自体が網羅性を担保できていないことがある。これを避けるためにKBO Treeを作成しビジネス/サービスが本来目指すゴールからブレークダウンし、採るべき戦略/やるべきこと(KBR)とその評価軸(KPI)について細部まで整理する必要が有る。

図:KBO Treeの例



各用語の意味:

- KBO(Key Business Objectives)/KGI(Key Goal Indicator):
  ビジネスにおける重要な目的/ゴール指標※定量
- KBR(Key Business Requirement):
  目標達成のための要求事項(戦略・戦術)
- KPI(Key Business Indicator):
  目標達成出来るかどうかを評価する指標※定量

※KBRとKPIは多段になる事があある。

参考:

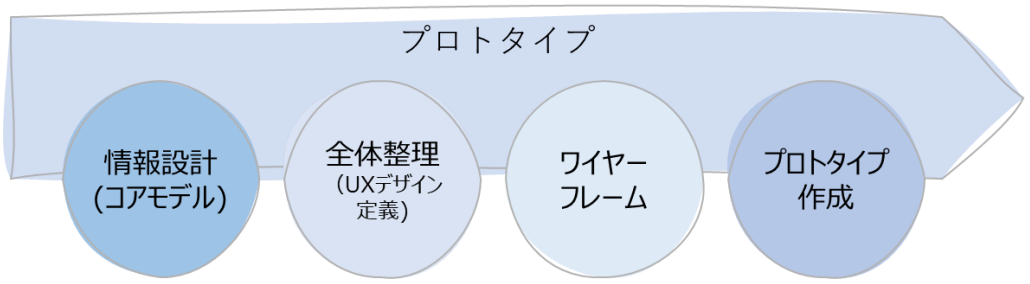
- 五足の靴:ゴールに近づくためのKPIじゃないと！
- KBO : Key Business Objectives Treeを作る
- 立場で違うKPI

- **プロトタイプ**

前段までの検討を踏まえてユーザーの課題を解決するために下記の様なプロセスで必要な機

能や情報・体験を定義し、サービスのプロトタイプを作成し今までの検討内容や仮説が適切であるかどうかを検証する。

図：プロトタイプのプロセス



a.　情報設計(コアモデル)＆全体整理(UXデザイン定義書)

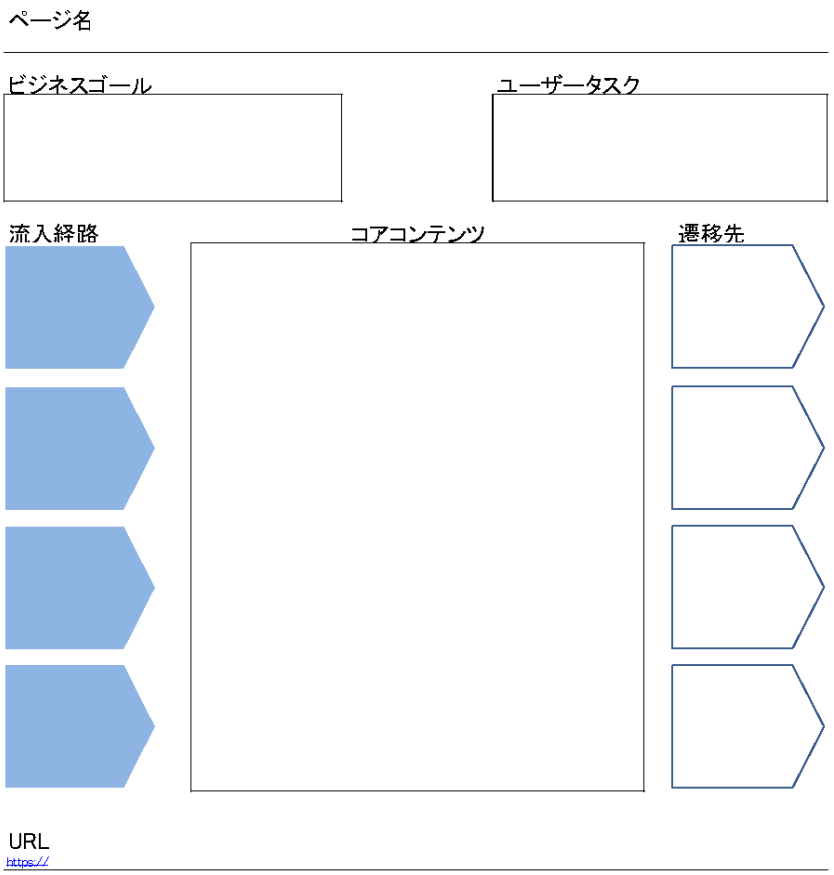
前段までの検討を踏まえてサービスのプロトタイプを作成する上で、
下記2つのドキュメントを用意していく。

- コアモデル
- UXデザイン定義書

・コアモデル　※必須

各ページにおける事業者側のゴールと来訪したユーザーのタスク、ユーザーが来訪/流入した経路、ユーザーが次にやるべきこと/遷移すべきページ、それを実現する為に必要な情報要素などを整理する必要がある。それらは、各ドキュメントにバラバラに纏められているかも知れないが、コアモデルという手法を使い1つのページ/画面単位で各段階において検討した内容を集約することで、各ページで醸成されたユーザーの理解や文脈について関係者が理解しやすく、有益な議論を行いやす状況を創り出す。

図：コアモデル



コアモデルは下記の様な順番で作成していく。
なお、作成にあたり、To-BeのCJMでタッチポイントとして定義されているページである必要がある。

1　ページ名を記載
ページの内容が分かるようなページの名前を付ける。ページに設定するタイトルなどでも構わないが、SEO的に全てのページが同じタイトルというのはNGなので、ページごとにユニークな命名を行うこと。
2　ビジネスゴール
事業者として、このページで達成したいことを記載する。なお、記載時はできるだけ具体的に達成したい事項を記載する。例)提供サービス群の把握および各サービスへのすみやかな送客
3　ユーザータスク
CJMにおいてユーザーがこのページで行うであろう行為やこのページにて知りたいこと等を記載する。
例）商品検討中なので商品について知りたい。毎月発生する料金について知りたい。
4　流入経路
このページに流入してくるチャネルを大まかに記載する。TVCMやリニューアルにより大幅に流入経路が変わる場合などは、変更する前提で経路を記載すること。
なお、重要な流入経路は背景色を赤色に変更して分かりやすくしておいておく。
例）自然検索、トップページ、サポートページより、、直接流入など
5　遷移先
To-BeのCJM上で定義されている遷移先および、その他ユーザーが遷移するであろう、遷移先を記載してください。なお、重要な遷移先は枠線を赤色に変更しておく。
6　コアコンテンツの記載

すでに設定済の項目であるこのページで事業者として達成したいビジネスゴールおよび、ユーザータスク、流入経路と遷移先を考慮して見出しレベルで構わないので、どういった情報/要素が必要なのか記載する。
また、対象のページについてボタンや検索枠などページの目的を達成するために必要な機能やインタラクションがすでに分かっている場合は、コアコンテンツの枠に必要な機能やその機能が有するインタラクション/反応等を記載する。
なお、記載においてはメインの遷移先、ユーザータスクを完遂出来る様に、コアコンテンツの掲載順番を決める。
※記載する際は内容が想定出来る程度に具体的な内容を記載する。
NG例)「見出し1」や「キャッチコピー」など
7　URLの決定
決まっている場合は、URLを記載する。
ドメインが未決定の場合はドメイン配下のディレクトリ階層およびページのファイル名が分かる様に記載する。

参考:
- The Core Model: Links and Resources
- UX MILK:Webライティングにおけるコンテンツ戦略
  ※ページ下部に記載があります。

・UXデザイン定義書　※あると便利
プロトタイプを開発する為に必要なユーザーの体験を整理する必要が有る。To-BeのCJMはユーザー毎の特定のユースケースにおけるユーザーのジャーニーとなっているため、全体を俯瞰するにはその都度全てのCJMやペルソナを確認しなければならず手間が掛かる。サービスやアプリ全体としてどのようなユーザーが存在し、それぞれがどう言った課題を持っているか？等について簡潔に纏め、関係者間で共通の認識を維持する必要があれば、「UXデザイン定義書」を作成すること。
これは、無くてもプロジェクトが止まるような致命的な影響がでないので、必要に応じて作成する。

作成方法及びテンプレートは以下から入手出来る。

参考:
- 「UXデザイン定義書」「アプリケーション定義ステートメント」に関する情報まとめ
- 株式会社ジェネシックス:UXデザイン定義書　※テンプレート

b.　ワイヤーフレーム
まず、コアモデルで定義したコアコンテンツに基づき見出し語、画像、テキスト、ボタンなど、一連の機能及び情報のかたまりを一つのブロックとしてページ内の各コンテンツの表示順番、表示位置、表示面積を設定し、各コンテンツとの関連性を考慮して、コンテンツ間の空白を調整した上でページ内に枠を配置する。
ワイヤーフレームを作成する際にはプロトタイプ開発に必要な要素については可能な限り正規の情報を入力する。

なお、作成方法や実際とサイトとワイヤーフレームの関係を確認するには下記サイトを参照のこと
参考
- Web担当者必見！ワイヤーフレーム作成時の注力ポイント9個と便利ツール20選
- Wireframe Showcase

c.　プロトタイプ作成
仮説に基づきコアモデルやワイヤーフレームを作成したが、設定した機能・コンテンツでTo-Be

のCJMで定義した体験をユーザーは得られるのか？と言う点について仮説を検証し、必要であればアップデートするサイクルを回す為にプロトタイプを作成する。
プロトタイプを作成する為には仮説の検証を可能にするために必要な要素が完成形ではないにしても検証可能なレベルで用意されていることが必要になる。

- ・デザイン手法
- ・ビジュアル・インターフェイスデザイン
- ・機能＆インタラクション
- ・パーツ/モジュールの設計
- ・画像・動画・テキストなどのコンテンツ

また、いきなりツールを使ってプロトタイプを作るのではなく、検討の初期段階において、紙と鉛筆を使いペーパープロトタイプ作成することが(以前)良くあった。ただし、ペーパープロトタイプはテスト参加者の『豊かな想像力』に依存してしまうため、デザインに対して直感的なフィードバックを集めるのは難しく、直接会った人としかテストが実施できないので、広範囲にテストを実施することは困難である。
さらに、詳細を検証するためにデザイン/機能プロトタイプに変換する必要があるが、変換には手間が掛かるので、現時点で使いやすいツールが市場に沢山存在していることを考えると、ペーパープロトタイプでの検証は止めて、様々ツールを活用しデザイン/機能プロトタイプを作成した方が効率的である。

参考:

- 手軽にアイデアを形にするペーパープロトタイピングのすすめ
- プロトタイプとは？ 3種類のプロトタイプとユーザー意見を反映させるメリット
- 【プロトタイプを作る！】初心者でもすぐに分かるProttの使い方
- プロトタイプの作成に役立つ！デザインツールInVisionの使い方【初心者向け】
- 2019年プロトタイピングツール8選～AIツールまで紹介【無料・ツール比較表あり】

・ テスト(ユーザビリティテスト)

仮説を検証する為に、プロトタイプを作成したが、それを適切に検証・評価・改善していく為に適切にテストを実施する必要が有ります。
ここでは、ユーザビリティテストを実施するために必要なテストシナリオ、テストの実施、テストで得られた結果の分析という観点で整理するが、
これ以外にも実際のサービス/サイト上で実施するA/BテストもしくはMVT(多変量テスト)等が存在する。

a. テストシナリオ作成

テストを実施するためにテスト被験者に対して口頭で説明するものである。テストにおいてユーザーがシナリオに基づいて無理なく自由に行動して貰うためにシナリオには複数の操作や自由さを内包したモノを用意する必要がある。
前段で定義されているユースケースやユーザーシナリオをそのままテストシナリオに転用した場合、テスト被験者の行動の自由度が低く、被験者はシナリオ内で設定されたタスクを黙々とこなし始め、適切なテスト結果を得ることができない。そのためユースケースやユーザーシナリオを元にテストシナリオを作成する場合は、ユースケースやユーザーシナリオの最初と最後だけもしくは最後(ゴール)だけを抽出し、設定された行動同士の間については課程を含めてユーザーが自由に行動できる余地を提供出来るように注意すること。

参考：
- ユーザビリティテストの進め方
- 調査目標からユーザビリティテストシナリオを作成する、7つのステップ
- ユーザーの目的をユーザビリティテスト用のタスクシナリオにしよう

b. ユーザビリティテスト実施

ユーザビリティテストはサービス・製品を評価するために実際のサービスの利用者や想定顧客と想定しているセグメントに存在するユーザーに試してもらいフィードバックを得るテスト手法である。

また、テストを行う場合は対象サービスの関係者全員が観察者として参加し、ユーザーの行動・言動・サービス上での反応を観察し、共通の課題認識が得られるように、多くの関係者の方に参加頂ける様に調整をすること。

また、参加した関係者が被験者の思考や行動に対してより深い理解を得るために被験者がどこを見て何を思っているか？について、『思考発話法』という手法を用いて、被験者が考えていることや受けた印象などについて制約を設けず自由に発言して貰う。
ただし、被験者はテストに慣れているわけでも、思考発話法になれているわけでもないので、テスト実施中はテスト推進側の人間が被験者に対して適切に声がけを行い、考えていることなどユーザーが感じていることを自由に発言できるように上手くコントロールする。
参考：
- Wikipedia：ユーザビリティテスト
- U-Site：ユーザビリティテスト
- UX/ユーザビリティのためのテスト – ユーザーテスト見学会 at JaSST
- ユーザビリティテストとは？基本からテストの種類まで詳しく解説
- ユーザビリティエンジニアリングをどう読み実践したか
- ユーザビリティテストをやってみよう
- 思考発話法 (think aloud) の注意点

c. 結果分析(インパクト分析)

テスト結果を分析・評価する為に下記の順番で問題点を洗い出し、分析を行う。

1. 問題リスト作成
   被験者が与えられたシナリオを完了するまでの間に、被験者が戸惑ったり、間違ったりしたところを、下記の様なフォーマットで纏める。
   これは問題点のとり残りを無くす意味で、観察者全員が作成する。

図：問題リスト

No.	被験者	タスク内容	画面	問題点	発話内容
1	被験者A	タスク1	ログイン画面	ログイン後に表示されるCAPTCHAで何をすれば良いか分からない	あれ？ログイン出来たと思ったんだけど、何か間違ったんですかね？
2	被験者B	タスク2	購入画面	商品購入に関する最終意思決定のボタンが探せてない	購入確認画面は表示されますが、、どこ押せば、買えるのでしょうか？すでに購入が完了しているのでしょうか？
3					

2. タスク達成状況

被験者がタスクを補助無く自力で達成できたかどうかを以下のようなマトリクスに纏める。

〇:完璧にできた、△:無駄な操作があった、×:独力でできなかった

図:タスク達成状況表

タスク	被験者A	被験者B	被験者C	被験者D
タスク1	〇	△	×	△
タスク2	〇	〇	△	×
タスク3	△	△	×	×

3. インパクト分析:

1. と2. の内容を元に「問題の質x発生頻度」で問題のランク付けを行う。

図:インパクト分析シート

	効果問題 →タスクが達成できない	効率問題 →タスクは達成できるが無駄な操作がある	満足度問題 →タスクは達成できるが不平・不満がある
発生頻度:高			
発生頻度:中			
発生頻度:低			

また、インパクト分析結果を基に改善を実施する場合は下記内容を参考の上優先順位付けを実施して改善を行っていく。

図:インパクト分析結果の優先順位付け

	効果問題 →タスクが達成できない	効率問題 →タスクは達成できるが無駄な操作がある	満足度問題 →タスクは達成できるが不平・不満がある
発生頻度:高	優先度:①	優先度:②	優先度:③
発生頻度:中	優先度:②	優先度:③	優先度:④
発生頻度:低	優先度:③	優先度:④	優先度:⑤

2. 参考情報

1) 共通して活用すべきフレームワーク

- ユーザー設定
  - セグメンテーション/ターゲティング
  - ペルソナ/プラグマティック・ペルソナ
- 課題把握
  - ユーザーシナリオ
  - カスタマージャーニーマップ(As-Is/To-Be)
  - サービスブループリント
  - バリュープロポジションキャンバス
- 仮説立案
  - KBOツリー(KBO/KGI、KBR、KPIがツリー構造になったもの)
- プロトタイプ
  - コアモデル
  - UXデザイン定義書
  - ワイヤーフレーム
- テスト
  - ユーザビリティテスト
  - インパクト分析

2) 共通して活用すべきITツール

プロトタイプ作成：AdobeXD / Prot / InVisionなど
※決めはないので好みで選択可

3) まとめるべきドキュメント

今までの項目を全て実施した場合、以下のようなドキュメントが生成されているはずである。

- ユーザーヒアリングシート
- ペルソナ定義書
- ユーザーシナリオ(ペルソナxシナリオの数だけ)
- カスタマージャーニーマップ
- サービスブループリント
- バリュープロポジションキャンバス
- コンテンツコアモデル(全ページ分)
- KBOツリー
- テストシナリオ
- インパクト分析結果

本来のサービスデザインでは上記ドキュメントに加えてビジネスモデルキャンバスやリーンキャンバスなども作成し、事業プランを整理、検証していく。
また、サービスをユーザーに提供する際のステークホルダーを整理するためにステークホルダーマップを作成し、スタッフ・ユーザー/顧客・パートナー企業・その他さまざまな利害関係者の役割や相関関係などが把握＆分析できる。
またビジネスモデルに基づき事業計画を立て、販売計画や投資計画よりROIや内部収益率(IRR)を算出し、投資対象として適切かを検証する。

参考：

- ビジネスモデルキャンバス(BMC)とは？ 作成方法やテンプレートを紹介

- リーンキャンバス(Lean Canvas)とは？ テンプレートや事例、書き方などを紹介！
- ビジネスモデルキャンバス・リーンキャンバスとは？分析を通して磨き上げられたビジネスへ
- ステークホルダーマップ
- Wikipedia：内部収益率法

参考文献

- UXデザインへの理解を深める～これからのデザイナーがすべきこと～
- Web担当者Forum：【仕事術】会議を飛び出せ! チームを一体化! 合意形成ワークショップのススメ
- So You Want to be a Service Designer – Jamin Hegeman
- 赤羽太郎 | サービスドリブンーサービスデザインが主導するイノベーション手法
- 「サービスデザインの時代」FITS2015

3.1.3 Business Process Re-engineering(BPR)

BPRの対象が国民・事業者等向けの行政サービスに関する業務である場合には、サービスデザインの観点から利用者中心のサービス設計としてBPRを実現すべきと考える。
ここでは、行政サービス以外の省庁内のバックオフィス業務を主眼としたBPRについて、経産省の幾つかのBPR事業に携わった経験から、課題と課題解決に向けて取り組むべき事項を記載する。

1．バックオフィス業務のBPRに関する課題

- **BPRの踏み込みが甘く、本質的な原因にたどり着いていない場合がある。**

＜発注者（経産省）側の原因＞
- ○ 省庁の業務には、民間企業にない業務や民間企業にあっても業務手順・ルールが異なる業務が多い。ところが業務フローや業務マニュアル・手順書、ルール等をまとめた資料が少なく、属人的な業務運用になっていて担当者に直接聞かないとわからない場合が多いため、事業者に業務詳細、課題点を十分に伝えきれていない。
- ○ 大きな変化に対する心理的な抵抗感を持つ職員や、法令・規則等や組織の論理を変えられないものとして当然視している職員も少なくないため、現状の制約を前提とした改善案の議論になってしまう。

＜受託事業者側の原因＞
- 省庁の業務に専門性があるコンサルタントがアサインされていない（そもそもそうしたコンサルタントは世にほとんどいない）。このため、あるべき姿がイメージできず、あるべき姿から課題を掘り起こせていない。
- 省庁のBPRや業務改革では、法令や規程等で規制されたルール、またはその解釈に基づいた運用ルールが障害になる場合が多いが、背景となる法令・規程等を精査し、職員と議論できるようなスキルセットを持ったコンサルタントがいない。このため、課題の本質的な原因分析ができず、経産省側の職員の言い分への反証を挙げることができない。
- 民間企業コンサルティング案件と同じスケジュール感で業務分析を進める傾向がある。このため関連資料の分析やヒアリング期間が短いなど業務把握のための時間が不足している場合が多い。ヒアリングした内容を十分に租借できないまま、課題を整理するため、本質的な原因追及に至らない。

- **部分的・局所的な改善検討に留まり、組織横断的・抜本的な改革になっていない。**

＜発注者（経産省）側の原因＞
- ○ METI DXとして位置づけられた事業であっても部門中心の検討（ボトムアップ型）になる傾向がある。このため、BPR／業務改革の検討範囲が部分的・局所的で、議論の過程で組織横断的な課題や原因を認識しても、検討範囲に含めず局所的な議論に留まってしまう。
- ○ 検討体制の中に組織横断的な課題に対する意思決定者がいない、エスカレーションパスがなくエスカレーションを行う術がない。担当者ベースで関係部門に協力を依頼するものの、しばしば全体効率よりも組織の論理や部門ローカルルールが優先され、本質的な課題解決に至らない結果になりがち。

＜受託事業者側の原因＞
- 発注者の要望に合わせた答えを準備しようとする御用聞き的な姿勢の事業者や、あえて事業範囲外の事項には触れないビジネスライクな姿勢の事業者が多いように感じる。スケジュール上の制約・工数的な制約があるため致し方ない側面もあるが、本質的

な原因分析を踏まえた上で、実際論を議論するような進め方になっていないように感じる。（スケジュール上の制約、工数的な制約は、その事業を企画する経産省側にも問題があるが、安　　値受注で工数余裕がない場合も見受けられる。）

- **BPR・業務改革が継続しない。**

＜発注者（経産省）側の原因＞

- ○ BPR／業務改革に着手しても、管理職や担当者が異動で替わるとトーンダウンしたり、方針が変わったりする場合がある。
- ○ 民間企業のようにBPR・改革の成果が企業／部門の収益に結び付く訳ではないため、継続するインセンティブが働かない。改善効果が見えづらいこともあり、小さな結果に満足してしまう。一つひとつの作業が楽に早く終わるようになっても、その分、別の作業が増えたり、元のやり方に戻ったり、改善の効果がどこかに消失してしまう傾向がみられる。
- ○ 数年単位のグランドデザインやゴールイメージを持たず、年度単位での業務改善を進める傾向がある。経産省として継続して取り組むべき事項に位置付けられないか、位置づけが弱いため、脈々と活動が継続される形になっていない。

＜受託事業者側の原因＞

- BPR定着に向けた提言ができてない。提言が抽象的・総花的なものに留まり、具体的なアクションをロードマップとして提示できていない。仕様書で求めていない場合もあるが、BPRを生業にする事業者ならBPR定着に向けた提言も当然に行うべきと認識。

2. 課題解決に向けて必要な取り組み

前述の課題解決に向けて、以下のような事項に取り組む必要があると考える。

（BPRの位置づけ・体制面）

- METIトランスフォーメーションもしくはMETI DXの方針のもとで実施する事業であることを改めて明確にし、BPR／業務改革を事業の目的・目標の主軸に定義する。報告・意思決定ルートにMETIトランスフォーメーションもしくはMETI DXを加えることで、部局内で判断ができない事項や、甘い判断になる事項のエスカレーションパスを確保する。
- 現状の業務に対して問題意識や課題認識がある職員を公募する形が理想。ただ現実的に難しい側面もあるため、選出されたメンバーに対して事前に個別面談を行い、BPRの目的・目標、メンバーに期待している役割をきちんと伝え、マインドセットした形で事業に臨んでもらうようにする。（BPRに対する取り組みや成果が人事評価できちんと評価されるように考慮することも必要。）

（事前準備）

- BPR対象業務の担当職員に、省庁の業務は民間企業とは異なった独特の業務と認識してもらい、事業者に業務内容が伝わるように資料を事前準備してもらう。（業務の流れ、作業スケジュール、関係する部門・役割、用語集などに加え、必ず業務に関連する法規・規程類も提示してもらう。）
- 事業開始前に、DXメンバー中心に対象業務関する事前ヒアリングを行い、各部門の課題事項を洗い出し、論点を整理しておく（ヒアリングの予行演習、手持ち解答の準備）。準備した手待ち解答は、事業者の課題整理の状況を見て事業者に提示するとよい。

（BPR検討時）

- 業務を把握する過程で、必ずその業務の根拠となる法令・規程類を押さえるようにする。そして法令・規程類が廃止・簡素化の阻害要因になるようなら、それらが法令・他省庁通達によるものか、経産省の内規・運用ルールによるものかを押さえ、改善策検討時にどのレベルのエスカレーションが必要になりそうかを掴んでおく。
- 事業のまとめとして、目指すべき姿、年度ごとの取り組み事項をロードマップとして整理し、担当者が変わっても実施すべき事項や改革の方向性が変わらないようにする。ロードマップでは、改善効果が出やすい部分、変化がわかる部分の取り組み事項を優

先して計画し、BPR／業務改革の流れが途切れないようにすることも肝要。
- BPR検討の過程で、事業の対象外の部分の原因分析が必要になった場合は、担当者の判断で検討対象から除外せず、必ずエスカレーションにより判断を仰ぐようにする。

（受託事業者との関係）
- 事業者には法令・規程類の読み込み、解釈を踏まえた原因分析は期待できないと考え、法令・規程類に起因する課題の原因分析は経産省側で実施する。（事業を担当する行政官の力量が問われるところ。）
- 現状把握のための時間を十分に取るように事業者に指示する（民間のプロジェクトに比べて、最低でも1.5倍程度の期間が必要）。経産省側は、ヒアリングの時間・回数に拘らず、例え原課の担当者から回数が多いとクレームが出ても、事業者が十分に理解するまでヒアリングを設定するか、個別に担当者の時間を貰うようにする。（ヒアリングとして公式の場を設定すると参加者が多くなる傾向があるため、二回目以降はピンポイントで個別に担当者の時間を貰う形がよい。）
- DXメンバー中心に経産省側でも目指すべき姿・あるべき姿を検討・整理しておく。ヒントなく正解を事業者に求めても期待した答えが出てこないか、時間がかかる場合が多い。タイミングを見て事業者に提示するようにする。
- 事業者には、ヒアリング結果をパワポやExcelにまとめることを期待しているのではなく、事業の目的外であっても本質的な原因分析と改革案の提示を期待していること、そしてその内容により事業者やコンサルタントを評価する旨を常に伝えるようにする。（通常のコンサルファームであれば、事業完了後、営業担当者やその事業の担当責任者が顧客の評価を聞きに来るはずなので、その際に、きちんと個別コンサルタントの評価結果をフィードバックする。）
- お役人の考えを綺麗な絵にすることが仕事だと捉えているコンサルンタントも少なくないと感じるため、経産省側の期待値や評価を提示し続けることが肝要。省庁のBPRに強い事業者があまりない状況を踏まえると、依頼した事業者やコンサルタントの評価を蓄積・共有し、依頼できるBPRの事業者を探すところからスタートせざるを得ない。

3. 参考情報

1) 共通して活用すべきフレームワーク

BPRのフレームワークには、シックスシグマ、BSC、4C、SWOT分析などのメジャーなフレームワークもあれば、それらのエッセンスをコンサルファームやコンサルタントがアレンジした独自のフレームワークがあるが、省庁のバックオフィスのBPRを進める場合のフレームワークとしては、ECRS（イクルス）の考え方が有効と考える。

<u>1. Eliminate(エリミネート) 排除（無くせないか）</u>

課題となっている業務の全部もしくはその一部を無くすことで、課題を排除できないかを考える。
前述のように省庁の業務は法令・規程類を根拠に執り行われるはずだが、長年の業務慣習で行っているものや、担当者や自部門の都合で付加されているものなど、その目的や法令根拠が明確でないものも少なくない。このため、必ず根拠となる法令・規程類を押さえ、法令根拠のない業務や作業は無くす方向で、経産省の内規・運用ルールが根拠になっているものは内規・運用ルールを変更し、その業務や作業を無くす方向で検討する。
他省庁所管の法令・規程類が根拠になっている作業は、所管省庁へ問題意識を伝え、意見交換等の場で改善の余地を探ると共に、経産省のしかるべきルートから改善の申し入れを行うべくエスカレーションパスに載せる。
法令が根拠になっている場合は、その業務や作業を無くすためには法改正が必要になるためハードルが高いが、諦めずエスカレーションパスに載せ、改善の声をあげるようにする。

2. Combine(コンバイン) 結合（一緒にできないか）

「排除」できない業務は、一つにまとめて処理できないか考える。METIトランスフォーメーションの一環でバックオフィス業務のBPOセンターの立ち上げを検討しているため、そこに巻き取ることができないか、部局単位で集中処理できないか等を検討する。

3. Rearrange(リアレンジ) 交換（変更できないか）

「排除」・「結合」ができない業務は、業務順序や担当者、場所などを入れ替えて、効率的に実施することはできないか、担当者や業務を実施する場所を変更することで、生産性を上げることができないかを考える。民間企業のBPRでは、この「変更」が有効な場合があるが、省庁では、単に他部署に業務を移すだけの検討になりがちで、「変更」だけでは効果が薄いため、次の「簡素化」と合わせて検討する。

4. Simplify(シンプリファイ) 簡素化（単純化できないか）

最後に、もっと簡単な方法で行うことができないか、やり方を標準化することで簡素化できないか、システムやツールを活用して効率的に作業できないかを考える。

2) 共通して活用すべきITツール

- 将来的には、BPMNツールを導入してもよいかもしれないが、職員が使いこなせないと事業者にメンテナンスを依頼することになり、効果が薄いように感じる。使い易いツールの選定、その有効性を実証してみる必要あり。

3) まとめるべきドキュメント

基本的に以下のドキュメントで十分と考える。

- 現行業務フロー
    - 経産省側で作成する形が理想だが、現実的には難しい。主要部分をDXメンバーで作成し、事業者に補完してもらう形が現実的。
    - 細かな業務パターンやプロセスを記述しても意味がない部分も多い。現行業務で課題になっている部分、改善効果が期待できそうな部分を詳細に記述し、他は網羅性を意識して整理すれば、詳細に記述する必要はないと考える。

- 新業務フロー
    - 現行業務フローと同様に、主要部分をDXメンバーで作成し、事業者に補完してもらう形にすれば、経産省側が求めていることが事業者に伝わりやすい。
    - 業務フローと言いながら、システムフローになっているケースも少なくないため、業務（ビジネスプロセス）を中心に記載するように意識する。資料は、基本的にExcelよりもPowerPointでまとめるようにする。（Excelの方が細く記述できるので、Excelを使用する事業者も少なくないが、逆にExcelだと細かく記述し、システムフローに近いものになってしまう傾向がみられるため。そもそもPowerPoint１～2枚でまとまらない業務はBPRが不十分と考えてよい。）

- BFC（Business Function Chart：詳細版、課題抽出版）
    - 業務概要、業務遂行上の課題、その原因、根拠法令・規程類、対応方針、業務担当者と作業工数（業務遂行に要する作業時間）を最低限記載する。特に「根拠法令・規程類」を明らかにすることが肝要。
    - SEに近いITコンサルタントが作成するとシステム中心の分析になってしまう傾向があるので、業務フローと同じく、あくまでもビジネス要件を中心に記載するようにする。

- SFC（System Function Chart：詳細版、システム開発一覧）
    - システム化を合わせて検討する場合に作成する。次の開発フェーズの調達仕様となる

ため、漏れがないように留意する。

- 目指すべき姿（ゴールイメージ）
    - DXメンバー中心に経産省側でも目指すべき姿・あるべき姿を整理しておく。事業者に意見や提案を求め、ブラッシュアップしていく。

- 目指すべき姿実現のためのロードマップ
    - 年度ごとの取り組み事項（各年度に誰が何を行うか）をロードマップとして整理し、担当者が変わっても実施すべき事項や改革の方向性が変わらないように心がける。

4) 参照すべき政府・経済産業省のルール

- 業務に関連する法令・規程類

（その他：参考図書）

- サービスデザイン実践ガイドブック（内閣官房情報通信技術（IT）総合戦略室）
- デジタル・ガバメント推進標準ガイドライン 実践ガイドブック「第4章 サービス・業務企画」（内閣官房情報通信技術（IT）総合戦略室）
- リエンジニアリング革命（M・ハマー＆J・チャンピー）
- 業務改革の教科書（ケンブリッジテクノロジーパートナーズ 白川 克・榊巻 亮）

3.1.4 要件定義・仕様書作成

要件定義・仕様策定においては、予算や工期を考慮し、目指すべき状態に対して、当該調達で目指す要件の範囲を決める必要がある。サービスデザインの各ステップで作成したドキュメントをもとに要件定義を行い、仕様書としてまとめる。

1. 要件定義・仕様書作成の課題

1) 機能要件

要件定義にはレベル感が重要である。インタフェースのようにパラメータも含めて厳密に要件に明示するもの、ユーザインタフェースのように機能を明示し、開発の繰り返し開発の中で最終仕様を確定させていくものがある。
また、サービスや技術が明確であり、要件として明示する場合と、開発者からの提案を期待して、概要の要件のみ記載する場合がある。
また、特定製品を想定し、要件に書き込むことにより他社の参入が制限されることもある。
よって、以下の観点から検討する必要がある。

- 要件の記述レベルは、想定している目的や開発方式に対して適正か
- 現在の記述レベルで、調達参加者は提案や見積もりが可能か

要件定義に起因する課題は以下のものがある。

- 要件定義が曖昧だったため、受託者が最低限のことしか想定しない
- 上記のため提案価格が安くなり、他社の目的に適合した提案が採択されなくなる
- 要件定義に対して、開発開始後に発注者が過大な要求をして開発が破綻する
- 要件定義が不足していて、実際に必要な機能の追加開発が必要になる
- 要件定義が過大であり、予算や期間内に開発できる提案者が現れない
- 要件定義が特定製品に依存していて、1者入札などになる

2) 非機能要件

非機能要件とは、入力機能とか表示機能などのように直接的に機能を表すものではなく、何秒で応答するとかセキュリティ対策をしているなどの機能を実現するにあたっての条件となる。
非機能要件は専門家以外にはわかりにくく、画面の応答時間はどのくらいまで許容できるかと担当者に問うと、「できるだけ早く」、「ストレルないレベルで」、「普通に」と抽象的なコメントが帰ってくることがあるが、非機能要件ではそこを数値で図れるようにする必要がある。また、セキュリティや運用の条件を厳しくすると、システム二重化が必要になったり、コストに直結する。その為、利用者の要求条件と費用を勘案し、要件化していくことが必要となる。
非機能要件の定義での課題は以下のものがある。

- 安心のために非機能要件を厳しく設定してしまい、予算がオーバーしてしまう
- 非機能要件が厳しすぎて、機能要件が実現できなくなる
- 非機能要件がわからないので、他の仕様書の非機能要件定義をそのまま使ってしまい、当該サービスに合わない非機能要件が設定されてしまう。

2. 課題の解決方針

1) 機能要件

課題の多くは、要件定義過程におけるコミュニケーション不足に起因するものである。政府調達においては、公平な調達を実施するために様々な制約がある。ベンダとのコミュニケーション方法も、RFI、パブリックコメントなどの意見照会、調達時の説明会、質問受付などの機会が設けられているが、短期間かつ柔軟に行うことが難しい。

そうしたことから、仕様書に参考資料を付帯するなど、ドキュメントベースでのコミュニケーションを充実させることが重要である。
参考の付帯資料としては、カスタマージャーニー、画面イメージ、課題一覧等、デザイン過程で作成された成果物が対象物になる。

アジャイル型の開発を行う場合には、機能要件には目的とした機能を記述する。アジャイル型開発と言っても開発が全く未定というわけではない、確定している機能を記載し、その何倍の開発規模を想定といったように、開発規模を明確にする。もちろん、発注者側に機能要件や開発規模を見られる人を入れることが重要である。開発途中で、専門家として開発実施の可否の判定を行うとともに、予算と開発実績・開発見込のバランスを取る酔にコントロールしていく必要がある。

2) 非機能要件

他の類似サービスの非機能要件を参考にする。また、非機能要件は専門性が高いため、わからないときには早めに専門家に相談することが重要となる。

3. 参考情報

1) 共通して活用すべきフレームワーク

政府システムの標準ガイドライン群が仕様書の雛形や各種情報を提供している。この情報をベースに作業をすすめる。

2) 共通して活用すべきITツール

要件定義は一般的なofficeツールで整備する。

3) まとめるべきドキュメント

要件定義書
上記を含んだ仕様書

4) 参照すべき政府・経済産業省のルール

政府標準ガイドライン群

3.2 サービスを形にする

3.2.1 調達

1. 総論

●契約の概要

国の会計法令においては、「調達」という用語は存在しません。会計法において国が国以外の組織から便益に対して対価を払う行為は、会計法29条「契約の意義」に規定がある。

会計法において各府省の長は、所掌に係る売買、賃貸、請負、その他の契約に関する事務を管理するとされていることから、財産価値の異動増減を伴う有償及び無償の契約があると考えられる。

また、売買、賃貸、請負、その他の契約に関する事務と記載されていることから、公法上の契約ではなく私法上の契約、いわゆる甲乙対等の契約であり、調達の担当者は、契約により目的に適った履行がなされ、かつ国にとって有利なものを提供できる事業者を契約相手方とすることが、納税者対する説明責任を果たす上で必要である。

国の契約については、公共の利益の増進を図る観点から、会計法は公正性、正確・厳正性、経済性の3つの原則を要請し、主として契約担当機関の行為について種々の規制を設けている。これらの規制は、原則として国の内部に対する制限である。

国の契約は、私人の一般の契約と同じように、契約の目的にかなった履行が確保され、かつ、有利なものとなる必要がある。このため、契約の相手方の選定は、適切に行わなければならない。

契約の意義（会計法29条）

各省各庁の長は、第十条の規定によるほか、その所掌に係る
売買、貸借、請負その他の契約に関する事務を管理する。

・売買、貸借、請負その他の契約」と規定していること
→財産価値の異動増減を伴う、有償双務契約のほか、無償契約なども含まれる。
・国の収入支出に関する手続を中心に国の財獲価値の異動増減の原因となる諸手続を規制する法規範であること

「契約」
国を一方の当事者とする債権関係の発生を目的とする合意

国の結ぶ契約には、公法上の契約（国が統治権の主体として相手方よりも優位な立場におかれる）と私法上の契約とがあり、また、金銭給付を目的とする契約と金銭給付を目的としない契約がある。本条の契約は、売買、貸借、請負等の私法上の契約で、主として国の金銭その他の財産価値の異動増減を伴うもの。

国の契約は、私人の一般の契約と同じように、契約の目的にかなった履行が確保され、かつ、有利なものとなる必要がある。
このため、契約の相手方の選定は、適切に行わなければならない。

（情報システム統一研修教材より）

●IT調達の位置づけ

デジタル・ガバメント推進標準ガイドラインの「第1章 標準ガイドラインについて　1. 背景及び目的」において、「今や政府情報システムは、単なる行政事務処理上の道具ではなく、行政運営の中核を成す基盤として存在するに至っている。さらには、デジタル技術は社会構造の変革の強力なツールとなっており、これまでの延長線上での改善ではなく、デジタル技術が国民生活やビジネスモデルを根底から変える、新しい社会が到来している。」と記載されている。

情報システムの調達は、ペンやノートのような消耗品の調達と異なり、「行政内部における行政サービスの利便性の向上並びに行政運営の効率性及び透明性の向上を実現するだけでなく、官民協働を軸として、行政サービスを改善」することが求められている。そのために既存の組織や制度、業務に対して、情報システムによるサービスや技術を用いることが必要である。そのため、調達はこのように既存の組織・制度及び業務に対して、情報システムのサービスや技術を結びつけ、新たな業務を実現するための手段と位置付けることができる。

（情報システム統一研修教材より）

●契約方式と競争性

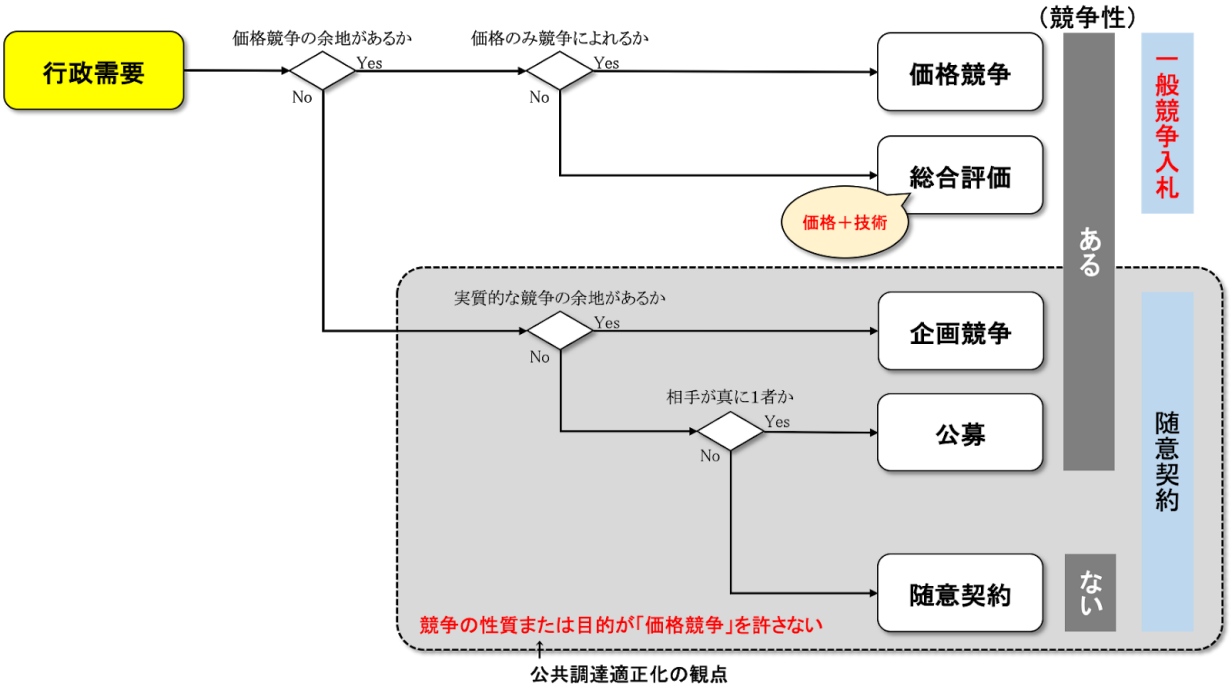
契約は、会計法第二十九条の三では、「契約担当官及び支出負担行為担当官（以下「契約担当官等」という。）は、売買、貸借、請負その他の契約を締結する場合においては、第三項及び第四項に規定する場合を除き、公告して申込みをさせることにより競争に付さなければならない」とされている。

契約方式には、一般競争入札と随意契約の2つがあり、一般競争入札には価格競争と総合評価、随意契約には企画競争、公募、随意契約の3つがあります。それぞれの概要は以下のとおりである。

契約方式	概　要
① 価格競争	会計法上の自動落札方式（最低価格を提示した者を落札とする）
② 総合評価	評価項目・評価基準を予め定め、評価点を決定し、入札金額と評価点との評価値が最も高い者を落札とする契約方式（予定価格の制限の範囲内）
③ 企画競争	予算限度額を設定し、その中で企画案を審査し、最も優れた企画案を提出した者を契約相手方とする契約方式
④ 公 募	請負相手方が、唯一者かどうか不確定の場合、目的、必要とする技術・性能等を明示して、HP上に広く公募し、応募者多数の場合は価格競争、総合評価もしくは企画競争へ移行。応募者が1者の場合は、唯一な者であることから随意契約を締結する契約方式
⑤随意契約	契約主体が契約の相手方を選定するのに競争の方法によることなく、任意に特定の者を選んで締結する契約方式をいう

●公共調達の適正化

公共調達の適正化について（財計第2017号　平成18年8月25日）では、「公共調達については、競争性及び透明性を確保することが必要であり、いやしくも国民から不適切な調達を行っているのではないかとの疑念を抱かれるようなことはあってはならない。
しかしながら、昨今、公益法人等との契約に関する各省各庁の運用には、広範囲にわたり、安易に随意契約を行うなど、必ずしも適切とはいえない事例があるのではないかとの指摘が行われるなど、国民に対する説明責任を十全に果たしているとはいえない状況となっている」との記載がなされている。これは、当時、IT関連事業の多重委託問題や分割少額随意契約問題、国土交通省・旧道路公団における橋梁談合問題、防衛施設庁官製談合問題、防衛施設技術協会、建設弘済会等との随意契約問題など度重なる不適切な調達により国民からの信用が失墜したことが契機となったところである。
このことから、「政府として随意契約の適正化について取組を進めた結果、競争性のない随意契約の見直しについての考え方が示されるとともに、今後取り組むべき課題として随意契約及び競争入札に係る情報公開の一層の充実等が盛り込まれた。これにより、「随意契約によらざるを得ない場合を除き、原則として一般競争入札（総合評価方式を含む。）による調達を行うもの」とされたところである。
これにより、従前は随意契約や指名競争入札などがなされてきたところであるが、調達の透明性を担保するためにも、企画競争や公募、随意契約に制限を設け、競争の性質や目的が「価格競争」を許さない場合のみに認められることとなった。



（情報システム統一研修教材より）

●失墜した信頼を回復するためには

「公共調達の適正化について」の契機となった不適切な随意契約や天下り法人への優遇にかかわらず、過去の調達においても度々信用を失墜する行為が行われてきたところです。その反省を踏まえて、公共調達においては、古くは会計検査院が設立されたり、会計組織が分化されるなどの改善が行われてきた。その改善が積み重ねられて、今の会計制度や調達手続きとなっている。
過去に不適切な調達を行ったとされる担当者の立場に立った場合、もしかしたら担当する業務の必要性や担当の考える正義のために行われた行為であったかも知れない。現に後に国に莫大な負債を背負わせることになってしまった「データ通信サービス」は、考案された当時は急速な情報システム投資が求められていたにもかかわらず、厳しい予算環境や国庫債務負担行為が認められない中で、事業者にリスクを寄せることで成立したビジネスモデルである。当時の予算審査の中では歓迎されたものの、ベンダーロックインや債務が積み重なったことにより、今では悪として評価されてい

る。
現時点で調達担当は「現時点では最善の窮余の策」とした方策も後に悪として評価され、更なる厳格な調達手続きが定められる契機となってしまうリスクがある。そのような将来の足枷とならないためにも、適切な手続きの遵守が求められる。
さらに、不適切な調達が積み重なったことによりマイナスのイメージが定着している公共調達だが、その汚名を雪ぐためにも単に手続きを適切に行うだけでなく、良いサービスをリリースすることで、情報システム調達は良い調達と認識してもらうよう努めなければならない。

言うなれば、

諸先輩が色々とヤンチャやらかした

信用失墜

＋　調達制度が面倒なのは明治以来の諸先輩のせい。。。

国民からの信頼を得るには
〇手続きにおける合規性の確保
〇調達に係る透明性の確保
〇実績の蓄積
→ よいサービスのリリース

手続きの遵守

（情報システム統一研修教材より）

【データ通信サービス】
データ通信サービス契約は、電気通信事業法に基づく電気通信事業者が、データ通信設備を用いて行う電気通信役務(ハードウェア等機賃貸、電力設備・建物提供、提供サービスに係るソフトウェア開発等役務)の提供契約である。
通常、長期継続契約を締結しており、本サービス利用の対価として、毎月、利用料金を支払うこととなる。
本契約を解約する場合、一定の支払額(契約解除金)が生じる可能性があり、この支払額がいわゆる「残債」とされた。
36のレガシーシステムのうち、社会保険オンラインシステム等10システムがデータ通信サービス契約を結んでおり、その契約先は全て(株)NTTデータとなっていた。
これらについて国会にて以下の決議がなされたところである。

【平成15年度決算警告決議】

3「社会保険オンラインシステム」に係るデータ通信サービス契約において、その経費・の積算の検証が不十分であったことは、誠に遺憾であり、また多くの府省のレガシー・システム等IT調達において、随意契約等による契約内容の不透明性など多くの問題が生じていること、加えて政府が当該調達にかかわる決算内容を把握していないことは、看過できない。
政府は、今後システムの、見直しを進めていく中で、不透明な契約内容の徹底的な見直し、汎用コンピュータのオープンシステム化、随意契約から競争契約への移行等の改善を図るとともに、当該調達にかかわる決算内容の検証・評価を厳正に行うべきである。

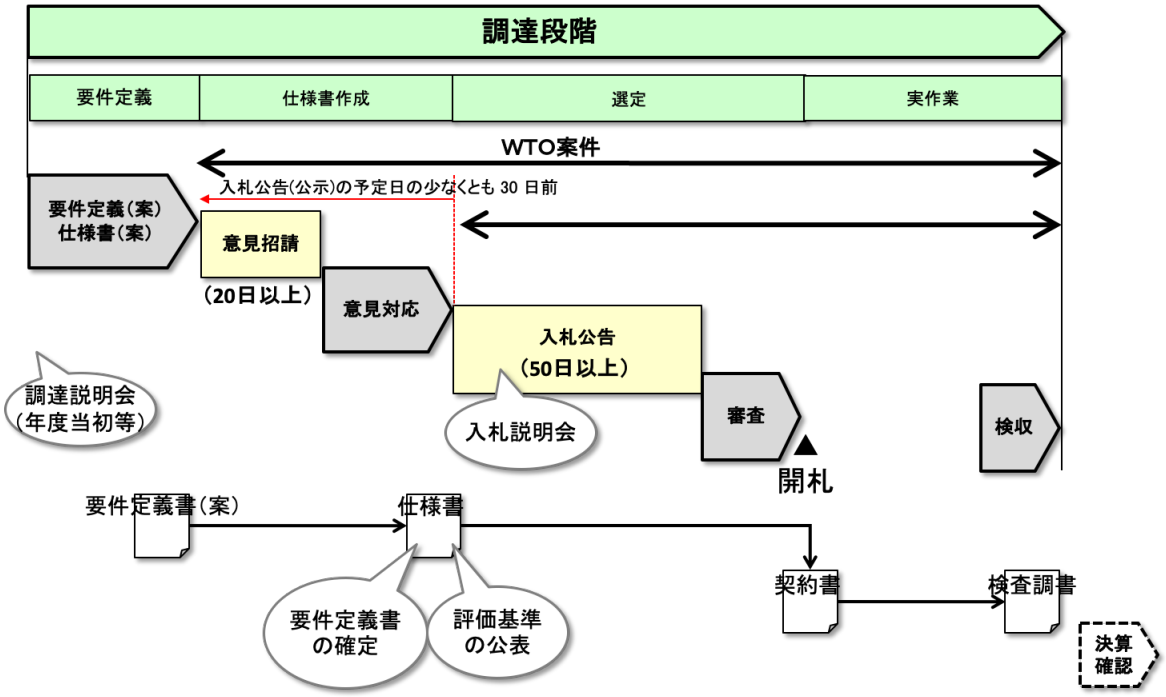
【平成 15 年度決算審査措置要求決議】

5 ITシステムの見直しについて
現在、政府部内には、厚生労働省の「社会保険オンラインシステム」を始めとするレガシー・システムが合計36あり、平成17年度予算で約3.572億円の経費が計上されている。
また、このほかに、「人事給与システム」や「災答管理システム」などの多くのITシステムがある。
これらのITシステム、とりわけレガシー・システムの調達は、技術的専門性の高さから、その多くがシステム事業者任せになり、システムの詳細がどうなっているのか、各府省側でその内容の把握が不十分なまま特定事業者との間で随意契約が繰り返され、その結果、不透明な契約内容、割高な契約額、システム事業者が開発したソフトウェアの著作権の帰属の実現、システム開発費を分割払としたことによる多額の残債の存在、政府全体として平成15年度におけるIT調達にかかわる決算額を確定することのできない事実等の多くの問題が生じている。
一方、会計検査院の「決算確認システム」では、その運用業務委託契約の業務内容を全面的に見直すとともに、契約方法を随意契約から一般競争契約へ移行させた結果、委託経費がそれまでの約30分の1に削減できたように多大の成果も見られる。
政府は、今後、レガシー・システムを含む77のITシステムについて「業務・システム最適化計画」を作成しシステムの全面見直しを進めていく中で、システム事業者が開発したソフトウェアの著作権の各府省への帰属の実現、システム開発費を分割払としたことによる多額の残債問題の解消等に努め、汎用コンピュータのオープンシステム化、データ通信サービス契約そのものの見直し、随意契約から競争契約への移行等の改善を図るとともに、当該調達にかかわる決算内容の検証・評価を厳正に行うべきである。
また、会計検査院は、以上の観点に留意して会計検査を実施すべきである。

なお、旧式(レガシー)システムとは中央省庁において、年間10億円以上の経費を要する情報システムであって、次のいずれかに該当するものである。
①汎用コンピュータ、オフコン(開発事業者独自のオペレーションシステムを搭載した中型コンピュータ)を使用したシステム及びこれらに接続するためのシステム
②平成6年以降、随意契約が継続しているシステム

●政府調達手続に関する運用指針

上述の倫理的観点に加えて、国際的な施策として政府調達については透明性、公正性及び競争性の確保が求められている。
1994 年4月15 日マラケシュで作成された政府調達に関する協定(以下「協定」という。)その他の国際約束に基づき透明性、公正性及び競争性の確保が図られてきている。供給者の利便及び競争力のある内外の供給者による市場参入機会の確保に資するとともに透明性、公正性及び競争性の高い調達手続とするため、協定及び2012 年3月30 日ジュネーブで作成された政府調達に関する協定を改正する議定書によって改正された協定(以下「改正協定」という。)その他の国際約束の対象となる調達及びこれに準ずる手続をとる調達について、我が国の調達においては、以下の運用指針に則ることとしている。
→運用指針詳細:政府調達の自主的措置に関する関係省庁等会議
ここにおいては以下のような事項が記載されており、国際社会の一員として守るべき事項となっている。
・10 万SDR未満の調達についても、透明性・公正性及び競争性を確保するよう努めるとともに、随意契約の縮減を図る。
・80 万SDR以上の調達については、年度開始又は年度開始前の可能な限り早い時期に予定される案件の公示を行う。
・80 万SDR以上の調達については、資料等の提供招請の公示の翌日から起算して少なくとも30日以降の日とする。
実際の調達手続においては、官報公示ための決裁に係る期間や国立印刷局における準備期間、意見招請に伴う意見回答期間、提案書審査の期間なども必要になってくるし、勤務日の関係もあることから、登庁日を考慮した実カレンダーにて手続きに必要な期間を管理することが求められる。



●関連根拠法令

〇会計法
〇予算決算及び会計令
〇契約事務取扱規則
〇公共調達の適正化について(財計第2017号　平成18年8月25日)
〇政府調達の自主的措置に関する関係省庁等会議
〇WTO政府調達協定

2. 入札公告

●根拠

〇入札の公告(予決令第74条)

第七十四条　契約担当官等は、入札の方法により一般競争に付そうとするときは、その入札期日の前日から起算して少なくとも十日前に官報、新聞紙、掲示その他の方法により公告しなければならない。ただし、急を要する場合においては、その期間を五日までに短縮することができる。

〇入札について公告する事項(予決令第75条)

第七十五条　前条の規定による公告は、次に掲げる事項についてするものとする。

一　競争入札に付する事項
二　競争に参加する者に必要な資格に関する事項
三　契約条項を示す場所
四　競争執行の場所及び日時
五　会計法第二十九条の四第一項の保証金(以下「入札保証金」という。)に関する事項

〇一般競争の公告(国の物品等又は特定役務の調達手続の特例を定める政令第5条)

第五条　契約担当官等が特定調達契約につき一般競争に付する場合における予決令第七十四条の規定の適用については、同条中「十日前」とあるのは「四十日前(一連の調達契約のうち最初の契約以外の契約に係る一般競争については、二十四日前)」と、「官報、新聞紙、掲示その他の方法」とあるのは「官報」と、「五日」とあるのは「十日」と読み替えるものとする。

2　予決令第九十二条の規定は、特定調達契約に関する事務については、適用しない。

〇一般競争について公告をする事項(国の物品等又は特定役務の調達手続の特例を定める政令

第6条）
第六条　前条第一項の規定により読み替えられた予決令第七十四条の規定による公告は、予決令第七十五条各号に掲げる事項及び予決令第七十六条の規定により明らかにしなければならない事項のほか、次に掲げる事項についても、するものとする。
一　一連の調達契約にあつては、当該一連の調達契約のうちの一の契約による調達後において調達が予定される物品等又は特定役務の名称、数量及びその入札の公告の予定時期並びに当該一連の調達契約のうちの最初の契約に係る入札の公告の日付
二　予決令第七十二条第二項の規定による申請の時期及び場所
三　第十条に規定する文書の交付に関する事項

●留意事項
競争は、せり売りに付する場合を除き、入札の方法により行わなければならない。（会計法第29条の5第1項）。入札の方法により一般競争に付そうとするときは、入札期日の前日から起算して少なくとも10日（急を要する場合は、５日まで短縮することができる）前に官報・新聞紙、掲示その他広く一般に周知できる方法により公告する（予決令第74条）ことが必要である。
入札者若しくは落札者がない場合又は落札者が契約を結ばない場合において、さらに入札に付そうとするときは、前記の公告の期間を５日まで短縮することが認められている（予決令第92条）。
入札について公告する必要事項は、競争入札に付する事項、競争に参加する者に必要な資格に関する事項、契約条項を示す場所、競争執行の場所及び日時並びに入札保証金に関する事項（予決令第75条）並びに条件違反の入札の無効（予決令第76条）及び契約書作成の有無（契約事務取扱規則第11条）だが、その他任意的公告事項として資材官給、違約の場合の措置、契約締結に要する費用等がある。
一般競争をする旨の公告の法的性格については、従前は学説も「契約の申込み」とするものと「契約の申込みの誘引」とするものと区々であったが、昭和36年の改正において「公告して申込みをさせることにより競争に付さなければならない。」（会計法第29条の3 第1項）とされたことにより、申込みの誘引であることが明らかになったところである。
申込みの誘引であることから、世の中に広く知らしめることが重要である。そのため、大型連休期間や年末年始、お盆の時期などに入札公告期間を設定するなどにより、入札公告について認知する機会を減らすことは法の趣旨に反することになる。

３．入札参加資格

●根拠
〇会計法　第二十九条の三
契約担当官及び支出負担行為担当官（以下「契約担当官等」という。）は、売買、貸借、請負その他の契約を締結する場合においては、第三項及び第四項に規定する場合を除き、公告して申込みをさせることにより競争に付さなければならない。
〇2　前項の競争に加わろうとする者に必要な資格及び同項の公告の方法その他同項の競争について必要な事項は、政令でこれを定める。

●競争に参加する資格
一般競争に参加する者の資格については、一般競争契約の性質上広く一般から参加できることとするのが最も望ましいのであるが、全く無制限に参加を求めた場合には、不信用、不誠実な者の参加により契約の履行が確保できないこととなるおそれがある。このため、一般競争を合理的かつ適正に行うためには、当該契約を完全に履行することができる能力を有することが必要であるとともに、契約について責任をもつ者でなければならないことから、従来から一般競争に参加させる者に一定の資格要件を設け、資格のない者については参加を認めないこととしている。これにより、以下の資格要件を定めている。
〇一般競争に参加させることができない者（予決令第70条）
以下に該当するものを参加させることはできない。
一　当該契約を締結する能力を有しない者
二　破産手続開始の決定を受けて復権を得ない者

三　暴力団員による不当な行為の防止等に関する法律（平成三年法律第七十七号）第三十二条第一項各号に掲げる者

〇一般競争に参加させないことができる者（予決令第71条）

一般競争に参加しようとする者が次の各号のいずれかに該当すると認められるときは、その者について三年以内の期間を定めて一般競争に参加させないことができる。

一　契約の履行に当たり故意に工事、製造その他の役務を粗雑に行い、又は物件の品質若しくは数量に関して不正の行為をしたとき。

二　公正な競争の執行を妨げたとき又は公正な価格を害し若しくは不正の利益を得るために連合したとき。

三　落札者が契約を結ぶこと又は契約者が契約を履行することを妨げたとき。

四　監督又は検査の実施に当たり職員の職務の執行を妨げたとき。

五　正当な理由がなくて契約を履行しなかつたとき。

六　契約により、契約の後に代価の額を確定する場合において、当該代価の請求を故意に虚偽の事実に基づき過大な額で行つたとき。

七　この項（この号を除く。）の規定により一般競争に参加できないこととされている者を契約の締結又は契約の履行に当たり、代理人、支配人その他の使用人として使用したとき。

〇各省各庁の長が定める一般競争参加者の資格（予決令第72条）

各省各庁の長又はその委任を受けた職員は、必要があるときは、工事、製造、物件の買入れその他についての契約の種類ごとに、その金額等に応じ、工事、製造又は販売等の実績、従業員の数、資本の額その他の経営の規模及び経営の状況に関する事項について一般競争に参加する者に必要な資格を定めることができる。

各省各庁の長又は、定期に又は随時に、一般競争に参加しようとする者の申請をまって、その者が当該資格を有するかどうかを審査し、資格を有する者の名簿を作成するものとすることになっている。

〇競争参加者の資格等を定めようとする場合の財務大臣への協議（予決令第102条の3）

各省各庁の長は、別に資格を定めようとするときは、財務大臣に協議しなければならない。

●留意事項

・民間事業者において与信管理を行うのと同様に、契約に責任を持てる者であることが必要なことから、入札参加には必要な資格が定められている。そのため、契約の種類やその金額等に応じて資格が定められる。

・入札参加資格が、定められた契約の種類や金額等に該当しない場合は財務大臣との協議が必要である。

・予定価格に応じた競争参加資格の等級区分は以下の通りである。

【物品の販売・役務の提供等】

予定価格			等級
3,000万円以上	～		A
1,500万円	～	3,000万円未満	B
300万円	～	1,500万円未満	C
300万円未満			D

4. 総合評価

●根拠

〇競争参加者の資格（会計法第29条の6）

2 国の所有に属する財産と国以外の者の所有する財産との交換に関する契約その他その性質又は目的から前項の規定により難い契約については、同項の規定にかかわらず、政令の定めるところにより、価格及びその他の条件が国にとって最も有利なもの（同項ただし書の場合にあっては、次に有利なもの）をもって申込みをした者を契約の相手方とすることができる。

〇契約担当官等が定める一般競争参加者の資格（予決令第73条）

契約担当官等は、一般競争に付そうとする場合において、契約の性質又は目的により、競争を適正かつ合理的に行うために特に必要があるときは各省各庁の長の定めるところにより、前記有資格者について、さらに必要な資格を定めることができる。これは、例えば、業者の手持工事の状況、技術者の有無その他の所有機械等の状況からみて有資格者全部を対象とすることが適当でない場合があるからである。各省各庁の長の定めるところによりとしているのは、契約担当官等の恣意を排除し公正な契約を確保するためである。

〇交換等についての契約を競争に付して行なう場合の落札者の決定（予決令第91条）

契約担当官等は、会計法第二十九条の六第二項の規定により、国の所有に属する財産と国以外の者の所有する財産との交換に関する契約については、それぞれの財産の見積価格の差額が国にとつて最も有利な申込みをした者を落札者とすることができる。

2 契約担当官等は、会計法第二十九条の六第二項 の規定により、その性質又は目的から同条第一項の規定により難い契約で前項に規定するもの以外のものについては、各省各庁の長が財務大臣に協議して定めるところにより、価格その他の条件が国にとつて最も有利なものをもつて申込みをした者を落札者とすることができる。

〇コンピューター製品及びサービスの調達に係る総合評価落札方式の標準ガイド（平成7年3月28日　調達関係省庁申合せ）

〇情報システムの調達に係る総合評価落札方式の標準ガイド（平成14年7月12日 調達関係省庁申合せ）

〇公共調達の適正化について（財計第2017号　平成18年8月25日）

〇情報システムの調達に係る総合評価落札方式の標準ガイドライン（平成25年7月19日調達関係省庁申合せ）

〇情報システムに係る新たな調達・契約方法に関する試行運用のための骨子（令和元年 5 月 29 日 各府省情報化統括責任者(CIO)連絡会議決定）

●留意事項

・総合評価の対象は平成7年のコンピュータ製品及びサービスの調達から、平成14年の情報システムに対象が広がり、平成18年の財務大臣通達により、さらにその範囲を広げてきたところである。

・総合評価は、価格及びその他の条件が国にとって最も有利なものをもって申込みをした者を落札者とする方式である。価格のみでの競争により難い場合に採用される。

・総合評価による技術的要件は、必須の要求要件及びそれ以外の要求要件に区分して、入札説

明書（仕様書を含む。）において明らかにすることが必要である。
・必須以外の要求要件については、総合評価基準において定める評価項目として評価の対象とするものに限るものとし、評価の対象としないものは記載しない。
・基礎点合計と加点合計との配点割合は、調達しようとする製品等の目的、用途等を勘案して適切なものとなるように設定するものとし、評価項目は調達上の必要性に基づき、「コンピューター製品及びサービスの調達に係る総合評価落札方式の標準ガイド」及び「情報システムの調達に係る総合評価落札方式の標準ガイド」別紙に掲げる項目から選択することとされている。なお、具体的な評価項目を設定する場合においては、その項目は当該調達に係る契約において、その内容が担保できるものに限るものとし、担保できないものは評価項目の対象としない。

【技術的対話による調達】

1．背景・目的
政府における一般的な情報システム等の調達においては、予算要求の段階から仕様を詳細に確定させることが困難な場合もあるため、行政と事業者とが一丸となって価値を生み出すためには、発注者たる行政と受注者たる事業者とが政策課題を共有し、対話を通じて相互理解を深めた上で契約することが重要であるが、調達仕様書等のドキュメント以外の意思疎通手段は乏しい状況にある。このような背景のもと、デジタル・ガバメント実行計画（平成 30年 7月 20日デジタル・ガバメント閣僚会議決定）及び未来投資戦略 2018（平成 30 年 6 月 15 日閣議決定）において、調達・契約方法の柔軟化について検討を進め、2020年度から試行的に開始するとされたことから、情報システム等の調達に「技術的対話」を取り入れた企画競争を試行的に実施することとしている。
※情報システム等の調達には、仕様書作成支援や工程管理などの情報システムに関連した役務調達も含む。

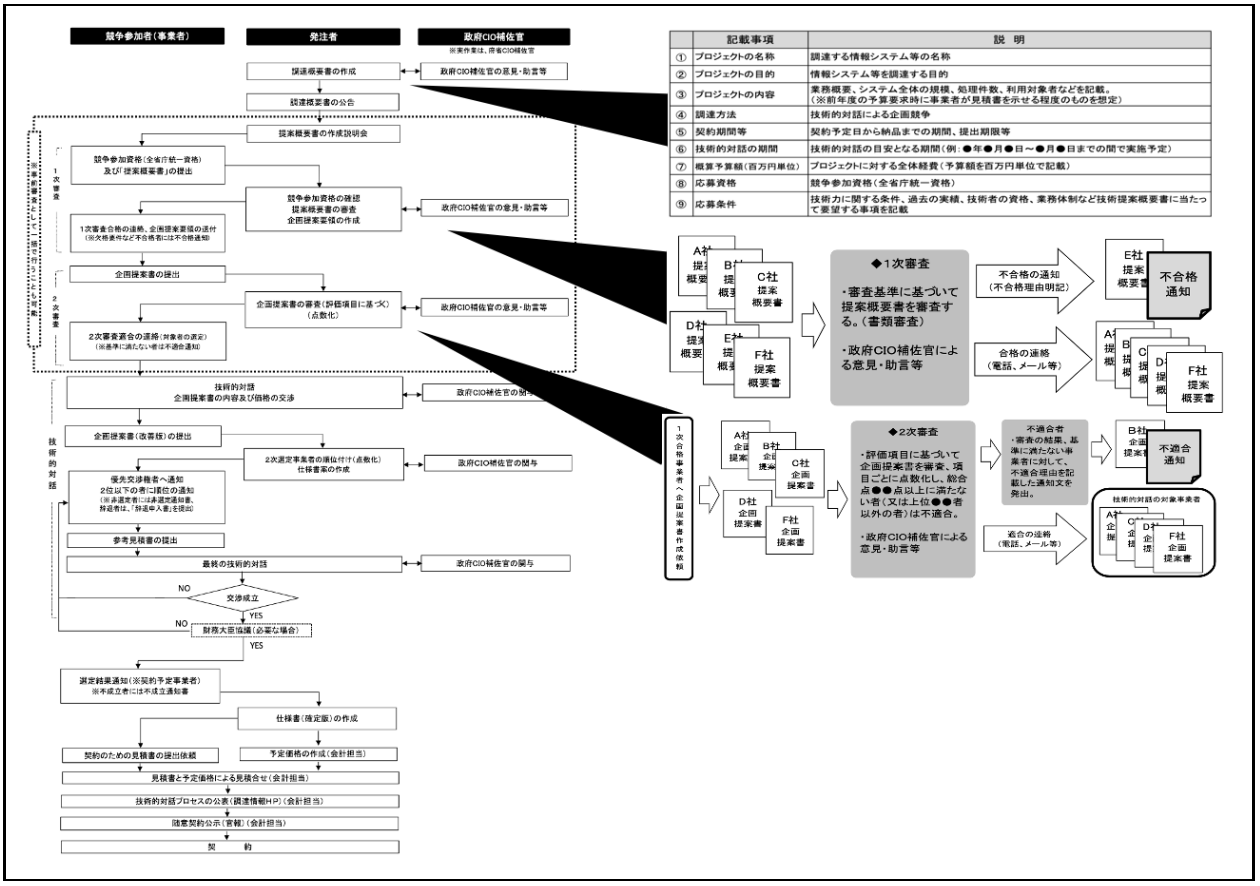
2．技術的対話について
技術的対話とは、発注者と事業者が対話によって、発注者が技術提案の改善・再提出を求め、事業者から技術提案の改善や価格の交渉を行う行為であり、これらを企画競争の調達手続の中で行うものである。

3．対象となる情報システム等の調達
適用対象となる情報システム等の調達は、「発注者が最適な仕様書を作成できない情報システム等の調達」又は「入札に付しても一者による応札が高いと想定される情報システム 等の調達」の2つの場合を想定し、具体的なイメージとしては、以下のいずれかの要件に該当する場合となる。

⑴「発注者が最適な仕様書を作成できない情報システム等の調達」
発注者側において最適な開発手法等の選択を行うことが困難であり、設計開発する事業者において、独自の高度で専門的な開発手法等を活用することが最適と考えられる情報システム等の調達
（具体的なイメージ）
・機微情報を取り扱うシステム及び漏洩による社会的・経済的混乱を招く恐れのある情報を取り扱うシステムの場合
・新規の調達であり、プロジェクトの計画段階で機能やリソースを確定させることは、政策目的を達成し得ないリスクが大きい場合
・大規模な情報システムの改修において、業務の実施方法や改修内容等の仕様の前提となる条件が複雑で多岐に渡るため、仕様書の提示が困難な場合
・技術的構造の異なる複数の情報システムとの連携や制度・業務見直しに伴う頻繁な機能改修が見込まれる場合

⑵「入札に付しても一者による応札が高いと想定される情報システム等の調達」
既存システム開発等の事業者による一者応札が複数年続いているシステムを対象に仕様内容等に関する技術的対話等を通じて条件の見直しを行うことで、複数の事業者の参加が期待され、かつコスト削減される可能性がある情報システム等の調達

4．技術的対話による企画競争に係る調達フロー



	記載事項	説　明
①	プロジェクトの名称	調達する情報システム等の名称
②	プロジェクトの目的	情報システム等を調達する目的
③	プロジェクトの内容	業務概要、システム全体の規模、処理件数、利用対象者などを記載。 （※前年度の予算要求時に事業者が見積書を示せる程度のものを想定）
④	調達方法	技術的対話による企画競争
⑤	契約期間等	契約予定日から納品までの期間、提出期限等
⑥	技術的対話の期間	技術的対話の目安となる期間（例：●年●月●日～●月●日までの間で実施予定）
⑦	概算予算額（百万円単位）	プロジェクトに対する全体経費（予算額を百万円単位で記載）
⑧	応募資格	競争参加資格（全省庁統一資格）
⑨	応募条件	技術力に関する条件、過去の実績、技術者の資格、業務体制など技術提案概要書に当たって要望する事項を記載

5. 契約管理

●根拠

〇会計法第29条の十一

契約担当官等は、工事又は製造その他についての請負契約を締結した場合においては、政令の定めるところにより、自ら又は補助者に命じて、契約の適正な履行を確保するため必要な監督をしなければならない。

〇監督の方法（予決令　第101条の三）

会計法第二十九条の十一第一項に規定する工事又は製造その他についての請負契約の適正な履行を確保するため必要な監督（以下本節において「監督」という。）は、契約担当官等が、自ら又は補助者に命じて、立会い、指示その他の適切な方法によつて行なうものとする。

〇監督職員の一般的職務（契約事務取扱規則　第18条）

契約担当官等、契約担当官等から監督を命ぜられた補助者又は各省各庁の長若しくはその委任を受けた職員から監督を命ぜられた職員（以下「監督職員」という。）は、必要があるときは、工事製造その他についての請負契約（以下「請負契約」という。）に係る仕様書及び設計書に基づき当該契約の履行に必要な細部設計図、原寸図等を作成し、又は契約の相手方が作成したこれらの書類を審査して承認をしなければならない。

●留意事項

・契約の適正な履行を確保するための監督を行う職員を監督職員といいます。監督職員は個人で任免されていることから、不適切な監督業務が行われた際には、監督職員個人が処分の対象になる。

・監督職員の職務は、履行に必要な細部設計図、原寸図等を作成したり、契約の相手方が作成したこれらの書類を審査して承認することにある。このことから、機能の不足や不十分な品質が生起した場合には、監督職員の職務が適切に履行されていないと言える。

６．検収

●根拠

〇会計法第29条の十一

２ 契約担当官等は、前項に規定する請負契約又は物件の買入れその他の契約については、政令の定めるところにより、自ら又は補助者に命じて、その受ける給付の完了の確認（給付の完了前に代価の一部を支払う必要がある場合において行なう工事若しくは製造の既済部分又は物件の既納部分の確認を含む。）をするため必要な検査をしなければならない。

〇検査の方法（予決令　第101条の四）

会計法第二十九条の十一第二項に規定する工事若しくは製造その他についての請負契約又は物件の買入れその他の契約についての給付の完了の確認（給付の完了前に代価の一部を支払う必要がある場合において行なう工事若しくは製造の既済部分又は物件の既納部分の確認を含む。）をするため必要な検査（以下本節において「検査」という。）は、契約担当官等が、自ら又は補助者に命じて、契約書、仕様書及び設計書その他の関係書類に基づいて行なうものとする。

〇検査職員の一般的職務（契約事務取扱規則　第20条）

契約担当官等、契約担当官等から検査を命ぜられた補助者又は各省各庁の長若しくはその委任を受けた職員から検査を命ぜられた職員（以下「検査職員」という。）は、請負契約についての給付の完了の確認につき、契約書、仕様書及び設計書その他の関係書類に基づき、かつ、必要に応じ当該契約に係る監督職員の立会いを求め、当該給付の内容について検査を行なわなければならない。

●留意事項

・給付の完了の確認を行う職員を検査職員といいます。検査職員は個人で任免されていることから、不適切な検査が行われた際には、検査職員個人が処分の対象になる。

・検査職員の職務は、契約書、仕様書及び設計書その他の関係書類に基づき、かつ、必要に応じ当該契約に係る監督職員の立会いを求め、給付の完了を確認することにある。このことから、機能の不足や不十分な品質が生起した場合には、検査職員の職務が適切に履行されていないと言える。

3.2.2 開発

1. 官公庁システムの開発における課題

官公庁におけるシステム開発では、民間と異なり、調達制度、事業執行、契約方法等の違いからいくつかの制約、課題が発生する。それらの課題制約に対する開発のあり方を記述する。

（官公庁システム開発の制約、課題）

a. 単年度事業原則の制約
   i. 毎年度調達（競争入札）
   ii. 単年度事業、および毎年度調達により短い開発期間
   iii. 開発と運用の分断
b. 入札制度の制約
   i. 入札は公平性を原則とし、開札まで落札ベンダーは分からない
   ii. 入札時のベンダー提案書の内容と実際のスキルの落差
c. 開発時の制約
   i. UIデザインについてベンダーのデザイナーに依存する
   ii. 開発プラットフォームはベンダーの提案に依存する
   iii. 行政職員とベンダーのITスキル差、業務知識差
d. 事業毎の独自仕様
   i. システム毎に統一されないデータモデル
   ii. APIの独自仕様

1-1. 単年度事業の課題

- 官公庁における事業は、基本的に毎年度予算要求を行い、予算要求を行った次年度に事業を執行する。（国庫債務負担行為による複数年度事業などの例外は除く）
- 事業執行にあたって外部業者に委託を行う場合、調達（入札）を行う。調達を実施するのは予算枠が確定する1月以降になるのが通例となっている。（予算枠確定前に調達実施はできない。）
- システム開発を行う場合、10万SDR（Special Drawing Rights, 1,500万円）以上の調達については政府調達の手続きに準拠して進めるが、意見招請、公告期間を行う場合、調達に3カ月かかるケースもある。
- 通常の事業は年度内で実施するが、前述のとおり、調達期間が必要となるため、システム開発期間は概ね半年～1年という短い期間で開発するケースが多い。
- また経済産業省では事業を単年度で実施することが原則となっていることから、初期システム構築とその後の保守運用が別事業となるケースもある。その場合、初期開発ベンダーと保守運用ベンダーが異なることも想定される。

1-2. 入札制度の制約

- 官公庁の調達は公平性を原則としており、一般競争入札を採用するケースがほとんどである。
- 一般競争入札で総合評価落札方式を採用すると、提案書に対する技術点および応札価格に対する価格点による総合評価で落札ベンダーを決定する。
- 事前の市場調査等で技術的に期待できるベンダーであっても、提案書の内容が伝わらないものであったり、価格において他ベンダーより高い場合、落札できないケースがある。
- 技術力が低いベンダーであっても、低価格かつ最低限の提案をするベンダーが落札するケースもある。

- また提案書の内容が期待以上に素晴らしいベンダーが落札しても、実際の開発が始まると期待した水準にないケースもある。
- 落札ベンダーは委託側では選択できず、技術水準も保証されないことから、プロジェクトを安定的に進めるためには委託側でベンダーをある程度コントロールすることが必要である。

1-3. 開発時の制約

- UIデザインはベンダーのデザイナーのスキル、感性に依存した成果物が提示されるため、期待したデザインにならないケースがある。
- 開発期間の短縮を実現するためSaaSサービスを提案されることがあるが、場合によってはベンダーの用意するSaaSプラットフォーム、サービスに依存するケースもあり、注意が必要である。
- 事業を担当する職員は必ずしもITスキルが高いわけではないため、ベンダーの提案するシステム構成等を判断するのは難しい。また、ベンダー側も行政業務に精通しているわけではなく、要件に関する認識漏れ、ズレが発生することがある。

1-4. システム毎の独自仕様

- ガバナンスが浸透していない状況では事業毎にシステムを開発すると他システムとの連携を考慮せず独自仕様で設計するケースがある。特にデータモデル（データベース）について本来標準化されるべき項目がシステム毎に設計が異なると、後々データ連携を行う際にデータ仕様の不整合に伴う変換機能を設けるといった非効率な開発が発生する。
- APIも同様で、データ型、データ長、繰り返し項目の持ち方等、後々のシステム連携を想定した設計がされていないとデータ連携のための追加機能が必要になる。

これらの課題について、入札時の仕様書等で予め防止することが可能な項目もある。以下、上述の課題に対し、仕様書に含める要件として参考になるプロジェクトの取組、意図しないベンダーが落札しても最低限プロジェクトを安定的に運営する取組を紹介する。

2. 課題の解決方針

2-1. 短い開発期間への解決方針

A. 調達前の開発期間の見積もりの精緻化

- 調達前の市場調査において、開発要件に対する必要な開発スケジュールを複数ベンダーに確認した上で、現実的な開発要件と開発スケジュールを設定することが重要である。
- 市場調査においてベンダーからより精緻なスケジュールを提示してもらうため、必要な作業、機能要件、非機能を整理して見積もりを依頼することが肝要である。
- 複数ベンダーに調査をする場合は民間見積り支援サービスWiseVine（https://corp.wise-vine.com/）を利用することで一括で確認できる。

B. クラウドサービスの利用

- システム開発は従来ハードウェア（インフラ）導入とアプリケーション開発に分類されることが多いが、ハードウェア導入で開発期間を取られることが多かった。クラウドサービスを活用することでハードウェア導入期間を大幅に短縮できる。
- SaaSの利用については、スクラッチ開発よりスケジュールを短縮できる可能性があるが、SaaSのコンポーネントや仕様等の制約を受けることもあり、要件に柔軟に対応できないケースも想定される。調達実施前に業務フロー策定、機能要件、非機能要件の整理を始めとした要件定義を精緻に行い、SaaSサービスと要件とのFit&Gap分析を行って適用可否を判断することが望ましい。事前に要件定義を実施できない場合でSaaSを選択する場合は、そのサービス仕様に制約を受ける場合があることをリスクとして想定しておく必要がある。
- クラウドサービスの利用については、政府CIOポータルの「政府情報システムにおけるクラウドサービスの利用に係る基本方針」（https://cio.go.jp/sites/default/files/uploads/documents/cloud_%20policy.pdf）を参考の上、選定することが肝要である。

C. 事業開始前の要件整理

- 事業開始前に発注者側で要件整理を実施することで、設計工程における検討期間を短縮できる。要件整理にあたっては、画面レイアウト、画面遷移、業務フロー、システムフロー、出力される帳票類が整理されていることが望ましい。サービスデザインで考えるとインセプションデッキ、ペルソナ、カスタマージャーニーマップ、ストーリーボード、ムードボード、サービスブループリント、ワイヤーフレームまで整理されていると開発に移行しやすい。（詳細はサービスデザインの章など参照）

D. 工程の並行実施（ファスト・トラッキング）

- 品質管理の質が下がるため積極的に採用するべきではないが、ベンダーが実施する総合テストと発注者が実施する受入テストを並行で実施することで最終確認工程を短縮することができるケースがある。ただし、単体、結合テストで品質が担保できていることを前提とする。
- なお、アジャイル開発手法は短期間開発の手法ではないので、期間が短縮できるというベンダーがいた場合は要注意。プロダクトバックログの積み上がり方次第では、開発期間が伸びるケースもある。

2-2. 初期開発ベンダーと保守運用ベンダーが変わるリスクへの備え

A. オープンな技術の採用

- 共通仕様書に記載があるが、中立性を保ったシステム構成を採用することが望ましい。また設計時にも中立性が保たれているか指摘して反映させることが望ましい。サードパーティサービスを採用する場合、可能な限り疎結合になるよう構成し、他のサービスに切り替えることが可能な構成を検討する。
- これまでのプロジェクトで見られたベンダー依存になりそうなサービス、フレームワークの例は以下のとおり。
    - SaaS内のコンポーネント（Salesforceの帳票ソリューション、フォーム作成ライブラリ、アンケート作成サービスなどサードパーティサービス）
    - ベンダー独自カスタマイズのJavaフレームワーク
    - ベンダーが独自カスタマイズして用意するクラウドサービス（〇〇 on AWSや、△△ on Azureなど）
    - CMS、チャットボット、FAQサービスなど
    - 監視サービス（受託ベンダーがベンダー内で共通的に利用するサービスを提供する場合がある。）

B. 移行作業の想定

- サードパーティサービスを使用せざるをえない場合、保守ベンダーが変わって異なるサービスに切り替わる可能性（ライセンスを購入できないため別製品を用意するなど）を考慮し、ソースコードやサービスで蓄積されるデータを標準的なデータ形式で納品してもらうようにすることが肝要である。
- ベンダーが切り替わるリスクを予め想定して、調達仕様の要件に移行作業を含めて記載しておくことが肝要である。
- 主な移行作業は以下の通り。
    - システム移行、サービス移行、環境移行（基盤の切り替え）
    - データ移行
    - ドメイン移行（DNS切替）
    - 運用移行（監視サービス、運用データの引継ぎ、コールセンター業務）
    - 開発環境、テスト環境移行
    - ソースコード移管
    - 開発ドキュメント移行

2-3. 落札ベンダーの技量に依存しないプロジェクト管理の方針

A. ツールによるプロジェクト管理の標準化

- ベンダーによって、課題の報告粒度、コミュニケーションの方法などプロジェクトを遂行する手法にばらつきが発生する。プロジェクト管理に慣れていない職員はベンダーの提案する方法が適切かどうか判断できない。ベンダー依存でプロジェクト管理を行う場合、旧態然としたExcelでの課題報告、メールによる連絡などは職員の確認に係る稼働も取ることになる。プロジェクトにおけるスケジュール管理、課題管理については、BacklogやJiraといったツールを活用することで、コミュニケーションロスやExcelの履歴を追いかけるといった無駄な作業を削減できる。

B. 品質と納期の管理

- プロジェクト管理の基本はQCD（Quality:品質、Cost:コスト、Delivery:納期）の管理であるが、発注者側は特に品質と納期に気を配る必要がある。
- 品質について、開発、テストが終わり、いざ受入テストを実施したところ、思っていた動作と違うということは往々にして起きる。主な要因は以下が考えられる。
    i. 要件定義、もしくは設計工程で発注者側がベンダーに要件を説明しきれていなかった。
    ii. 要件定義、もしくは設計工程でベンダー側が要件を理解できていなかった。
    iii. 正常系の処理は十分認識合わせできていたが、異常系の処理の考慮が検討できていなかった。
    iv. 業務フローの検討に不足はなかったが、データのバリエーションを考慮できていなかった。
- i、iiについてはテキストベースで要件を確認している場合に発生するケースが多い。要件定義工程、もしくは設計工程であってもモック、プロトタイプなど、より完成時点のイメージを共有できる方法で要件を確認することが望ましい。iiiについて、厳密に進めるのであればIPAの要求逸脱分析表（https://www.ipa.go.jp/files/000043910.pdf）を参考に異常系を検討する。HAZOP（http://jasst.jp/symposium/jasst12tokyo/pdf/D2-1.pdf）も参考になる。
- 納期について、基本的に受託ベンダーはプロジェクト計画書を策定する時点において、要件に基づいた実現可能なスケジュールを提示する。スケジュールが遅延するのは概ね以下の要因が考えられる。
    i. 発注者起因のスコープ変更
    ii. ベンダーの見積もりの甘さ
    iii. システム構築時の想定外の不具合
    iv. 外部要因（外部協力者の都合、災害の発生など）

- ウォーターフォール開発で上述のスケジュール遅延はよく見られる。発注者側は要件

定義工程、もしくは設計工程で責任を持って要件を確認する必要がある。プロジェクト内の責任者だけでなく、組織の責任者とも要件の認識を合わせて、後工程でのスコープ変更を減らす活動をすることが肝要である。
- 合意したスコープはドキュメントに残すことが重要である。
- アジャイル開発の場合、トレードオフスライダーでプロジェクトの優先事項（スコープ、コスト、スケジュールのどれを優先するか）を関係者内で合意して置くことが肝要である。
- トレードオフスライダーの優先事項に基づいて、要件をトリアージし、プロダクトバックログに落とし込むことで、現実的な開発スケジュールを確保する。

2-4. UIデザインの担保

A. ガイドラインの適用
- 省庁事業などにおけるサービス間で共通的なデザインで構築することは、サービスのブランドイメージの構築にとって重要である。
- 共通的なデザインの構築にはデザインシステムの適用が効果的であるが、実装レベルで定義するためにはプラットフォームまで指定する必要がある可能性もある。デザインシステムの構築が難しい場合は、最低限ガイドラインで遵守すべき事項を定義し、トンマナを揃えることが肝要である。
- 経済産業省、および政府全体のガイドラインは以下の通り。

〇広報室ガイドライン
https://newintra-hp/qqbbbl/120625intra/web/index.html#guideline
・スタイルガイドライン（ウェブサイトの運用・管理規定）
・ウェブサイトデザインガイドライン（ウェブサイトの基本ルール）
・ウェブアクセシビリティガイドライン（JIS X 8341-3:2016 適合レベル「AA」準拠目標）

〇政府CIOポータル
https://cio.go.jp/guides
・Webサイト等の整備及び廃止に係るドメイン管理ガイドライン
・政府情報システムにおいてサービス提供の対象とすべき端末環境及びWebブラウザの選定に関する技術レポート
・Webサイトガイドブック

〇みんなの公共サイト運用ガイドライン
https://www.soumu.go.jp/main_sosiki/joho_tsusin/b_free/b_free02.html#bf2-0

B. 外部コンサルタントの活用
- 受託ベンダーのデザイナーにデザインを任せると、意図しないデザインになるケースがある。そのような状況を抑止するため、発注者側のデザインの意図を汲み取れる外部デザインコンサルタントを用意し、デザインに関するレビューに参加して指摘してもらう手法もある。

C. クラウドソーシングの活用
- 受託ベンダーのデザイナーにデザインを任せるのではなく、クラウドソーシングを活用して、複数のデザイン案から優秀なデザインを採用する手法がある。クラウドソーシングでデザイン案を公募する場合、デザインに求める要件（明るい、まじめ、ポップなど）を予め整理するとともに、遵守するガイドラインがあれば提供する。

2-5. 標準ガイドラインの適用

A. 標準データガイドブックの利用
- 法人に関するデータモデルについて、データガイドブックが策定されており、当該ガイド

ブックの内容を適用することで、複数システム間でのデータモデルを標準化することが可能になる。

B. APIガイドブック、データ連携モデルの適用

- 政府CIOポータルでAPIガイドブックが公開されており、APIの実装方式について規定されている。当該ガイドブックに沿った設計を行うことで他システムが利用しやすいAPIの構築が可能になる。

3. 開発事例

3-1. クラウドサービスを活用した開発期間の短縮

事例A) 中小企業庁　認定経営革新等支援機関電子申請システム

事業概要:

- 中小企業の経営、行政手続を支援する支援機関を経済産業省が認定する制度において、電子申請、審査および認定結果の公表を行うシステム
- 年間1万件の申請を想定

システム構成:

- Salesforce

開発期間(初期開発):

- 2018年9月～2019年6月(約8カ月)

要件定義、設計	2カ月
プロトタイプ開発	1カ月
開発	3カ月
テスト	2カ月

3-2. 初期開発ベンダーから保守ベンダー変更時のトラブル

事例B) 中小企業庁　経営力向上計画電子申請プラットフォーム

事業概要:

- 中小企業が指針に基づいて経営力を向上する事業計画を提出し、認定されると税制優遇等を受けられる制度において、電子申請、審査および認定を行うシステム
- 年間1万件の申請を想定

システム構成:

- Salesforce

開発期間(初期開発):

- 2018年9月～2019年3月(約6カ月)

発生した課題：

- 初期構築を行ったベンダーが社内規定の変更により、保守期間のSalesforceのライセンスを提供できなくなり、保守事業から撤退
- Salesforceの契約組織を保守ベンダーに譲渡できる前提で保守事業を入札にかけたが、初期開発ベンダーより譲渡できない旨申し出あり。

実施した対処：

- 保守ベンダーにてSalesforceの組織を再構築、データ移行を実施
- 再構築およびデータ移行の間、システムは停止（約2カ月）
- 再構築およびデータ移行のコストは保守ベンダー負担（開発スコープを削って捻出）

3-3. Backlogの活用

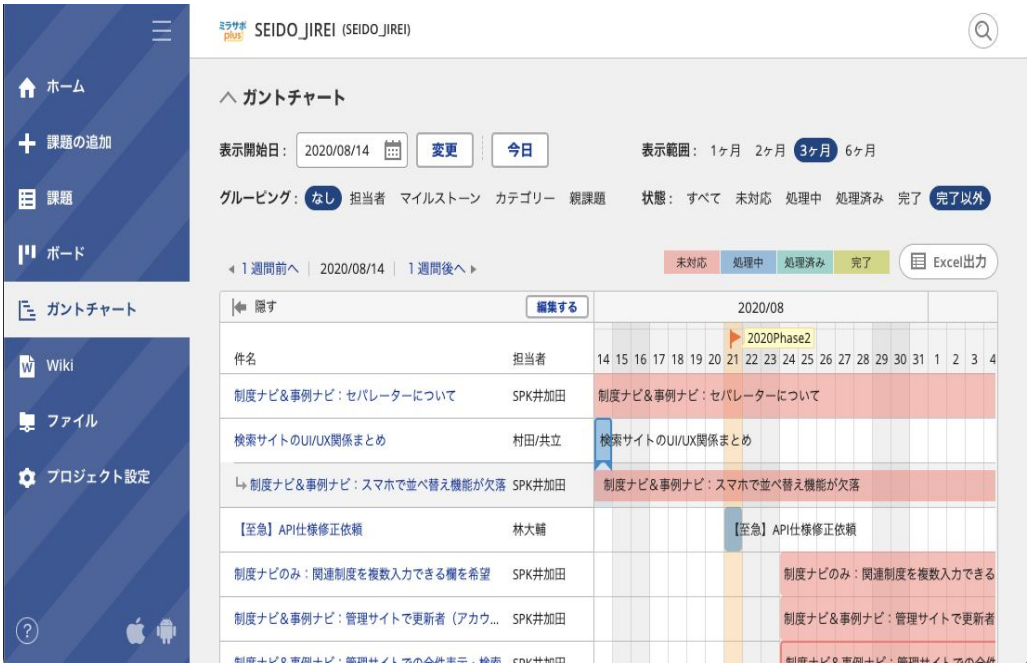
事例C) 中小企業庁　認定経営革新等支援機関電子申請システム

活用イメージ：

①定例会の進行

- 定例会のチケット作成して、議題となるチケットを列挙
- 定例会の中で解決すれば、その場で結論を入力してクローズ
- 優先度の高い課題をまとめて確認できるので効率的
- 議論を各チケットのコメントに記述することで議事録作成稼働を削減

②ガントチャートによるスケジュール管理



- マイルストンを設定して、リリース項目を管理
- チケットをステータスで管理し、遅延チケットを把握して必要な対処を検討

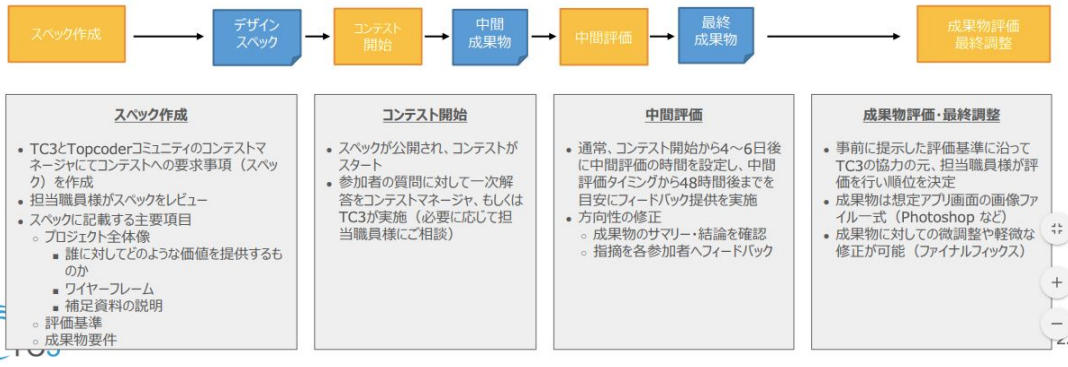
3-4. クラウドソーシングの活用

事例D）中小企業庁　事例検索サービス（事例ナビ）プロトタイプ構築
事業概要：

- 中小企業が各種支援制度に紐づく事例を検索するサービスのプロトタイプを構築
- UIデザインで世界的コミュニティトップコーダーにコンテストを開催して複数案から選定

（1）コンテストの進め方

- デザインに求める要件を反映したコンテストスペックを作成し、コンテストを実施した。
- 中間チェックポイントを設け、中間レビューを実施した。
- 最終成果物に対する評価を実施し、最終成果物を獲得した。



- 詳細は以下の報告書に記載
  - https://www.meti.go.jp/meti_lib/report/H30FY/000041.pdf

3-5. デザインガイドラインの適用

事例E）中小企業支援プラットフォーム
事業概要：

- 中小企業庁が構築したサービスにおいて、経済産業省のホームページデザインに準拠したスタイルガイドに基づいて共通的なトンマナ、ヘッダー構成、フッター構成を採用

経済産業省ホームページ

ミラサポ plus

スマートSMEサポーターサイト

経営力向上計画プラットフォーム

3-6. デザインコンサルの採用

事例F）中小企業庁　ミラサポplus、事例ナビ、制度ナビ
活用内容：

- 諸般の事情により、分割調達することになったミラサポ plus、事例ナビ、制度ナビのデザインを統一的なUI、UXにするため、各事業の受託ベンダーとは別にデザインコンサルを小額随意契約にて採用

デザインコンサルの役割：

- 各事業のUIレビューに参加し、統一的なデザインとなるようベンダーが提示したデザインに対してレビューを実施
- 最終成果物が意図したデザインとなっていることを確認

3-7. 標準データモデルの採用

事例G）中小企業庁　中小企業支援プラットフォーム
活用事例：

- 申請データのワンスオンリーを見据えたデータ項目の共通化を図るため、A票（法人基本情報）、B票（申請個別情報）、C票（法人財務情報）のフォーマットを参考にデータモデルを設定
- A票、B票、C票はデータガイドブックとして再整備された。

4. 参考情報

1) 共通して活用すべきITツール

01. プロジェクト管理ツール
    a. Backlog
    b. Jira
    c. Redmine
02. プロトタイプ作成ツール
    a. Adobe XD
    b. figma
    c. inVision
    d. Prott
    e. Marvel
03. 設計支援ツール
    a. Lucid chart
    b. EnterpriseArchitect

c. Astah
d. PlantUML
e. Salesforce Dev Tools（Salesforce利用時のみ）

04. ドキュメント管理
a. SharePoint
b. Backlog
c. Confluence

05. コミュニケーションツール
a. Backlog
b. Slack
c. Teams

2) まとめるべきドキュメント

A. ウォーターフォール開発
a. ユースケース図
b. 業務フロー
c. システムフロー
d. 画面レイアウト
e. 画面遷移図
f. 機能一覧
g. シーケンス図
h. クラス図
i. データベース設計
j. ER図
k. バッチ処理設計
l. システム構成図、ネットワーク構成図
m. 環境定義書（パラメータシート）
n. 運用設計

B. アジャイル開発
a. 仮説キャンバス／リーンキャンバス
b. インセプションデッキ
c. トレードオフスライダー
d. ペルソナ
e. カスタマージャーニーマップ
f. サービスブループリント

3) 参照すべき政府・経済産業省のルール

A. プロジェクト管理
a. デジタル・ガバメント推進ガイドライン（政府CIOポータル）
b. デジタル・ガバメント推進標準ガイドライン実践ガイドブック（政府CIOポータル）
c. サービスデザイン実践ガイドブック
https://cio.go.jp/node/2421

B. システム設計
a. コード（分類体系）導入実践ガイドブック
b. API導入実践ガイドブック
c. APIテクニカルガイドブック
d. 政府情報システムにおいてサービス提供の対象とすべき端末環境及びWebブラウザの選定に関する技術レポート
e. Webサイト等の整備及び廃止に係るドメイン管理ガイドライン

C. Webデザイン
    a. Webサイトガイドブック（政府CIOポータル）
    b. 経済産業省スタイルガイドライン
    c. ウェブサイトデザインガイドライン
    d. ウェブアクセシビリティガイドライン

D. データモデル
    a. マスターデータ等基本データ導入実践ガイドブック
    b. 行政基本情報データ連携モデル
    c. 行政サービス・データ連携モデル

E. オープンデータ
    a. オープンデータ基本方針（政府CIOポータル）

3.2.3 テスト・デプロイ

■テスト

1. プロジェクト体制の三権分立化

プロジェクトにおいて、テストチームはプロダクトマネージャーや開発チームと同等の立場であることが望ましいべき。三者が疎に独立した上で、それぞれの責務の明確化と情報連携、お互いをチェックし合う体制を構築する。プロジェクトのキックオフに端を発して、あらゆる活動に適切にテストチームが参画することで、テスト活動を円滑に進めることができる。重要なことは、発言権や決定権を持ちつつ協業と分業を行うことである。

- ●課題
  - テストチームは、プロジェクト内の他チームと比較して発言力が低い、情報が共有されにくいといった立場上の課題があり、これが下流工程でのテストフェーズにおいて様々な問題をもたらすことが少なくない
  - プロジェクトの初期フェーズにおいてテストのプランニングがなされないままプロジェクトが進行する場合、テストフェーズへの十分な議論や検討がなされず、テストリソースの不足やテストスケジュールにおけるフィージビリティ漏れや、テスト設計の抜け漏れによりるテスト実施の遅延を発生させ、結果的にリリース遅延やリリース後の重大な障害を発生させてしまう
  - テストプランやリリースプランを見直す機会を持たずにテストフェーズに進んだプロジェクトの要件/設計フェーズ場合や、実装開発フェーズで遅延が発生している場合は、テスト期間を圧迫することとなるため、結果的にテスト不足、リリース後の重大な障害を発生させてしまう
- ●チェックポイント
  - ビジョンやミッション、バリューがテストチームに共有されること
  - プロジェクトの計画フェーズでテスト計画（フィージビリティやリソースなど）について議論が行われること
  - 三者各々が作成するドキュメントをレビューし合うプロセスが明文化されていること
  - 要件や機能の変更や遅延の情報が適切なタイミングでテストチームに共有され、必要な議論が行われること
  - テストチームはプロダクトの品質担保においてプロフェッショナルであり（あるべき）、この点においては権限と責務を持つこと
  - 組織が内製化されているかどうかで、導入の難易度に違いがでること
- ●解決方針
  - プロジェクト内の他チームと比較して、テストチームの立場が低いことを問題の主因と捉え、プロジェクトの体制をプロダクトマネージャー、開発エンジニア、テスターで3権分立化した上で初期フェーズ（例えばキックオフ）からテストチームが参画する
  - 3者間での責務の明確化と情報連携、互いのチェック機構を構築する
- ●ベストプラクティス
  - プロジェクトの体制を以下のようにロール毎に責務を明確化した上で三権分立化し、キックオフからテストが参画する
    - プロダクトマネージャー　:プロダクトの要求仕様に責務を持つ
    - 開発エンジニア　:プロダクトの技術仕様、実装に責務を持つ
    - テスター　:プロダクトの品質担保に責務を持つ
  - 三権分立化した上で、一者のドキュメントに対して他二者のリード担当者がレビューを行い、いつ誰が承認したかを明確に記録できるようなサインオフプロセスの導入を行う
  - ドキュメントには合意形成した事を示す各自のサインを記入する欄（サインオフテーブル）を用意し、必須のサインオフが全てチェックされることでドキュメントがFixされ、次のフェーズやタスクに進むことができる
  - 以下にサインオフテーブルの例を記載する

サインオフテーブル

- ご自身の名前の欄に、レビュー完了日とサインチェックをお願いします。
- サインチェック欄は編集モードで'x'を記載するとチェックがつきます　※そのままチェックしても保存されません
- アサインさせて頂くレビュアと、レビューして頂く観点は以下となります。 ※敬称略

レビュア	レビュー観点	レビュー完了日	サインオフ	コメント
@inagaki (METI)	テストチームでの内部チェック	2019/10/24	☑	問題なし。
[redacted]	開発に関連するテスト開始基準やトリアージプロセス、テスト範囲やシステム構成など	2019/10/29	☑	
[redacted]	ユーザー要件に関連する目的や範囲、スケジュールなど	2019/10/29	☑	

- サインオフ後にドキュメントの変更が発生した場合は、再度サインオフプロセスを実施する

●実践検証

- jGrants1.0（補助金申請システム）の開発事業においては、当初よりテストチームが開発チームの下に位置付けられており、すでに実装開発が終盤に差し掛かりいよいよテストフェーズに進む段階で、リリースに間に合わないほど進捗が逼迫していた背景より、最小限の変化で最大限の効果が出るようテストチームのみ半独立化を行った
- 体制上の位置付けには変更はせず、実働として開発チームと切り離し、分業と協業を行ったことでテストドキュメントにおけるサインオフプロセスやバグトリアージプロセスを容易かつ効果的に導入することができた
- 結果として、プロダクト品質・テスト品質どちらも担保しつつ期限内にテスト完遂、リリース完遂の一翼を担うことができた
- 一方で、プロジェクト後半からの突貫的な対応であったことから、完全な三権分立はできておらず様々な課題は残ったものの、jGrants2.0の開発事業に繋がる検証ができたと言える

2. テスト活動に必要なドキュメント

プロジェクトにおいて、ドキュメントの必要性は言うまでもないが、一方でむやみにドキュメント化することにより、情報が煩雑化・分散化したり、無駄な作成・保守コストが増加するといった問題が発生する。これらを解消するためには、最低限の数や量で効果的なドキュメントを、できるだけ早く効率的に作成する必要があり、どのような種類のドキュメントをいつ作成するかはステークホルダー間で事前に取り決めておくことが重要である。

テスト活動は、要件や技術仕様、開発内容などから様々な影響（特に悪い影響）を受けるといった特性上、ドキュメントにおいても比較的頻繁に変更を余儀なくされることが多い（テストケースが顕著な例と言える）。これらの影響を防ぎつつ、ステークホルダーと円滑な合意を取れるドキュメント体系を検討する必要がある。

●課題

- テストフェーズの営みに関する情報を記載したドキュメントをステークホルダーがレビューするタイミングが遅いと、テストフェーズ開始までに実施すべきタスクを完了できない
- テストケース/シナリオは膨大な情報量になるため、プロダクトマネージャーにとってプロダクト要件が漏れなくテスト要件に落とし込まれているかをレビューすることが困難である
- いつ誰がどのような観点でレビューし合意形成できたかが不明なままテスト実施を進めてしまうと、プロダクト品質、テスト品質の劣化を招く
- 繰り返し実施する可能性のある優先度の高いテスト要件やケース/シナリオがわからず、リグレッションテストや自動化テストで効率よく実施すべきスコープがわからない

- テストに関わるドキュメントがプロジェクトのステークホルダー誰からでもいつでも容易にアクセスできるように管理できていないと情報の抜け漏れが発生する

●チェックポイント

- 業務要件が策定されたタイミングからテスト要件化が可能であること
- テスト要件がプロダクト要件をカバーできていること
- テスト要件に優先度をつけること
- 機能要件/非機能要件が策定されたタイミングからテスト設計化が可能であること
- レビューすべきドキュメントが各ステークホルダーの観点に合わせた情報粒度となっていること
- テストの進捗状況や発生課題、結果をできる限り早くかつ正確かつわかりやすく共有すること
- テスト後の残課題、残タスクを次に繋がるよういつでもアクセス可能な状態に記録しておくこと

●解決方針

- テストドキュメントに対して、プロダクトマネージャーと開発が各々の観点からレビューを実施し、合意形成されたらドキュメントにサインオフする(ドキュメントへのサインオフプロセスの導入)
- 以下にサインオフテーブルの例を記載する

サインオフテーブル

- ご自身の名前の欄に、レビュー完了日とサインチェックをお願いします。
- サインチェック欄は編集モードで'x'を記載するとチェックがつきます　※そのままチェックしても保存されません
- アサインさせて頂くレビュアと、レビューして頂く観点は以下となります。 ※敬称略

レビュア	レビュー観点	レビュー完了日	サインオフ	コメント
@inagaki (METI)	テストチームでの内部チェック	2019/10/24	☑	問題なし。
[redacted]	開発に関連するテスト開始基準やトリアージプロセス、テスト範囲やシステム構成など	2019/10/29	☑	
[redacted]	ユーザー要件に関連する目的や範囲、スケジュールなど	2019/10/29	☑	

●ベストプラクティス

- テスト活動に必要なドキュメント体系を以下とする
  - テスト要件書(計画書)
    業務要件を基に、テストゴールやテスト要件、テスト優先度やテストスコープ概要、テスタビリティ、テスト方法概要やテスト環境、体制やスケジュール、リスクなどの情報を記載する
  - テスト設計書
    機能要件/非機能要件を基に、テストスコープ詳細やテスト方法詳細などの情報を記載する
  - テストケース/テストシナリオ
    テスト設計書の下位ドキュメントとする
    テスト実施の入力条件やパターン、期待結果、ステータスを記載する
  - テストデイリーレポート
    テストの日々の進捗や結果、検知した不具合の情報を記載する
  - 振り返りドキュメント
    テスト活動全般におけるKPT&Aをメンバー間で議論、共有する
  - テスト完了レポート
    テスト実施後にテスト結果やバグトリアージ結果を記載
    必要に応じて、KPT&Aで整理した今後の施策を記載する
- レビュアーにとって粒度が細かく情報量の多いテストケースを全てレビューすることは

非効率であるため、テストケースと上位設計ドキュメント（テスト設計書）を分離し、テスト設計書のみレビューを実施する
    - テストケースのフォーマットはテストの自動化を見据えて、人による可読性と自動化ツールへのインプットデータとしての構造化の両要件を満たす統一フォーマットとする
    - テストケースはドキュメントファイルとしてではなく、データ資産としてGitなどのリポジトリでバージョン管理する
- **実践検証**
    - jGrants1.0（補助金申請システム）の開発事業においては、上記ベストプラクティスを全て実践した
    - 同プロジェクトには明確なプロダクトマネージャーは存在しなかったが、開発事業者側にプロジェクトマネージャーがいたため、各テストドキュメントを作成し、プロマネや開発チームにレビュー依頼、合意（サインオフ）を取ることで、テストフェーズにおける認識齟齬をほぼ防ぐことができた
    - 一方で、テスト実施中に開発側がテストチームへの事前共有なくデプロイやデータ削除などを行うなどテスト品質が担保できない状況が発生したが、テスト要件/計画ドキュメントのリスク項目から漏れていたことに起因するため、今後テスト実施前にドキュメントを通じて意識を合わせておくことが必要と感じた

3. テスタビリティの追求

テスタビリティ(Testability)はテスト実行可能性（テスト容易性）であり、テストが十分に実施できるか（テストが実行しやすいか）の観点から指標化されることが望ましい。テスタビリティが高いことにより、不具合を見つけやすいという利点が生まれる。一方で、テスタビリティが低いことで、テストで不具合を見つけるためのコストが高くなったり、そもそもテストが実行できない（するべきであるにもかかわらず）、といった問題が発生するため、テスト実施の前工程にて明確化しておく必要がある。
なお、テスト自動化を導入する場合にはテスタビリティを高めておくことは必要不可欠である。

- 課題
    - テストデータやテスト環境の考慮不足により、テストが開始できない、テストが止まってしまうなど実施が困難となる
    - ログやデータベース、権限付きの画面などにテスターがアクセスできないと、テストが開始できない、テストが止まってしまうなど実施が困難となる
    - 要件や仕様などのドキュメントの情報からテスト要件を抽出できないと、テストの設計や実施が困難であったり抜け漏れが発生する
- チェックポイント
    - テストドキュメントを容易に作ることができること
    - テストを円滑に実行できること
    - テストの出力内容、エラーや不具合を確認できること
    - テスト操作時の制御ができること
    - テスト対象の範囲を分割して確認しやすいこと
    - テスト対象の構成がシンプルであるかどうかを把握できること
    - システム変更の実行や頻度が把握できること
- 解決方針
    - テスタビリティ観点を持ち、テストに関連するドキュメントや情報へ指摘を行う
- ベストプラクティス
    - 機能要件/非機能要件のレビュー時に、テスタビリティの観点で不足する情報についてフィードバックする
    - コードやデータベース、設定ファイルなどのシステム構築要素を直接参照することなく、ログやデータ、設定情報を確認することができるように可視化ツールを導入する
    - 不具合を検知した際に、テスター自身が原因切り分けの調査を行い、バグチケットに記載して、プロダクトマネージャーや開発リードにエスカレーションする
    - テスト自動化ツールを導入する前に、テスト対象のシステムをアーキテクチャレベルで

テスタビリティの高い設計にしておく

- ●実践検証
  - ○ jGrants1.0（補助金申請システム）の開発事業においては、すでに実装開発が終盤に差し掛かりテストフェーズに進む段階であったため、テスタビリティ観点でステークホルダーと議論する余地がなかった
  - ○ プロダクト要件をレビューする段階でテスタビリティを確認することが望ましい

4. バグトリアージと優先度

リリース前にバグを0件にしなければいけない、という考え方は改める必要がある。全てのバグを治すコストは膨大であるにもかかわらず、その費用対効果は大きくない。それよりも治すべきバグを把握し、適切に対処する方が価値が大きい。そこで、治すべきバグはどれかをできるだけ迅速に判別するための手法がバグトリアージである。

- ●課題
  - ○ 検知したバグのステータスや情報の可視化ができていない場合、バグ管理が困難となったり、ステークホルダーとのコミュニケーションコストが高くなる
  - ○ 検知した全てのバグをリリースまでに改修することは現実的に避けるべきであり、各々のバグに対してリリース判定基準を元にした対応の優先度をつけて高い順に実施していくべきであるが（バグトリアージ）、それを行わずに手をつけ易いバグから闇雲に対応していくと、期間内に優先度高のバグを改修することができなくなったり、想定外の潜在バグを内包してしまうリスクを抱える
  - ○ テスト期間中に検知したバグに定期的に優先度をつけずに期間の後半にまとめて対応する方法を採用した場合、優先度高かつ対応作業量高かつ回避策なしと認識されたバグはリリースまでに対応することができず、結果的にリリースを延期することとなる
- ●チェックポイント
  - ○ トリアージという語源を理解し、時間をかけることよりも迅速に処理することに重きをおくこと
  - ○ プロダクトマネージャーが主体となって、開発リードやテストリードと共にバグトリアージを実施する体制であること
  - ○ テスト期間中にタイミングを決めて（例えば毎日夕方1回など）バグトリアージを実施すること
  - ○ バグに優先度をつける前提として、業務要件に優先度がついていること
  - ○ できるだけバグは治さない（ワークアラウンドへ誘導）という判断を行うこと
  - ○ トリアージ（判断）の記録を残しておくこと
  - ○ 後に誤った判断であることがわかった場合は、判断理由の見直しと改善を行うこと
- ●解決方針
  - ○ 検知した不具合をチケット管理ツールで可視化し扱いやすくする
  - ○ トリアージプロセスを導入する
- ●ベストプラクティス
  - ○ テスト実施期間中に検知したバグに対して、早急に対応の要否を判断し仕訳する作業をプロセス化（トリアージプロセス）する
  - ○ トリアージの観点は主に、「ユーザーへの影響度」「発生頻度」を指標として用いることが多く、2つの軸の高低の組み合わせからリリースまでの対応の優先度を決定するのを基本形とする
    ※影響度が高くても発生頻度が低い場合は優先度低となる場合もある
    ※上記2軸以外に、フィージビリティやコストなども指標となる
  - ○ 優先度(Priority)は以下のようにPxなど順序がわかりやすい表現を用いることが望ましい
    - ■ P0 ：緊急対応レベル
    - ■ P1 ：必須対応レベル
    - ■ P2 ：可能な限り対応レベル
    - ■ P3 ：対応不要レベル

- ○ 優先度を低くトリアージしたバグについては、対応が容易であっても優先度が高いバグよりも先に改修してはいけない
  ※改修することでさらに優先度の高いバグを生むリスクを発生させるため
- ○ テスト実施完了時点で優先度が高いバグが1件でも残っている場合、運用フェーズでのワークアラウンド（回避策）が存在しない限りは、リリース判定時のプロダクト品質判断としてはNGとなる
- ○ P2, P3はできるだけ対応しない(Won't Fix)という判断にすることが望ましい
- ○ 検知されたバグは前後述のバグトラッキングシステム(BTS)でチケット管理することで、トリアージを効率良く進めることができる

●実践検証

- ○ jGrants1.0（補助金申請システム）の開発事業においては、上記ベストプラクティスを全て実践した
- ○ テスト期間中、毎日定時頃を開始時間とし、日次で検知した不具合チケットをMETI(PO)と事業者(テスター)で全てトリアージし、優先度とコメントをつけて開発チームに毎日共有した
- ○ 不具合の検知数や複雑性によってはトリアージは長時間行われることもありトリアージ実施者にとっての負担は大きかったが、より早いタイミングでバグへの対応が行えたため、テスト終盤に大きな問題が発生することはなかった
- ○ 特筆すべきは、トリアージプロセスの導入によって、全てのバグを治す必要がない（治すべきでない）というマインドをプロジェクトメンバーが意識することができ、取捨選択が行えたことである

5. クオリティゲートの設置と効果

総合テスト/QAテストの実施に耐えうるプロダクト品質でない場合に、不具合が多発し、調査・修正対応にコストがかかることから、テスト遅延のリスクが高い。
実装完了後に開発チームからプロダクトやモジュールが引き渡された際に、どの程度の品質であればテスト開始可能であるかの判断基準としてクオリティゲートを設定する。実施されるテストの種類やプロジェクトの品質保証のレベルに応じてクオリティゲートの高低が決められ、その基準に該当する基本的なチェックレベルのテストを迅速に実施し、全てパスして初めて本格的なテストが開始される。パスできなかった場合は開発チームに差し戻されることとなる。
クオリティゲートにより、本来テストすべきではないレベルのプロダクトに時間をかけてテストしたりバグを調査したりといったコストを削減する効果がもたらされる。なお、クオリティゲートを効果的に運用するにはチェックテストを自動化することをおすすめする。

●課題

- ○ 開発から引き渡されるプロダクトの品質が低い場合、テスト開始後に重度のバグが大量に検知されるため、テスト進行の妨げ（テストブロッカー）となり予定期間内に完了できなくなる

●チェックポイント

- ○ 開発チームのテストに依存しないよう、予めクオリティゲートのレベルをステークホルダー間で合意しておくこと
- ○ 実施前にテスト設計（テストケース/シナリオ含む）が完了していること
- ○ テスト要件で付与した優先度を元に、クオリティゲート用のケース/シナリオをピックアップすること
- ○ いつ実施するかを事前に決めておくこと
- ○ クオリティゲートレベルは自動化できるように定量化されていること
- ○ 実施結果をどのようにレポートするかをステークホルダー間で合意しておくこと

●解決方針

- ○ ステークホルダー間で合意の元、クオリティゲートを導入する

●ベストプラクティス

- ○ 開発実装フェーズ（単体テストや結合テスト含む）が完了し、テストチームが実施するレベルのテストフェーズ（総合テストやシステムテスト、QAなど）を開始する前（テスト開始

日の前日など）に実施する
    - 以下にクオリティゲートレベルの例を記載する

クオリティゲートレベル

テスト計画書 6.テスト開始基準より抜粋

レベル	レベル説明	詳細説明
Q1	支障なくテスト実施できるプロダクト品質を満たしている	基本シナリオテストとシナリオ内の各機能パターンテストでP0, P1レベルのバグが検知されない
Q2	テスト実施にあたり懸念はあるが、緩和策を立てればテストを実施できるプロダクト品質を満たしている	基本シナリオテストでP0,P1レベルのバグが検知されない
Q3	テスト実施できないプロダクト品質である	基本シナリオテストでP0,P1レベルのバグが一つでも検知される

    - 上記では、Q1,Q2はパス、Q3はフェイルとなり開発チームへ差し戻し対応後にリトライすることとなる
    - テストケース/テストシナリオに優先度が付与できていれば、クオリティゲート用にピックアップするコストが軽減できる
    - 一般的にテスト粒度はスモークテストレベルで充分であり、できる限り時間をかけずに開発側へフィードバックすることが望ましい
    - テスト自動化を導入している場合は、クオリティーゲートも自動化することが望ましく（繰り返し実行されうるテストであるため）、さらにCI/CDパイプラインと連携すれば、開発側がテスト環境に自動デプロイした直後にクオリティゲートを自動的に実行/レポートできるため、一連の流れにかかる時間を大幅に短縮することができる
- **●実践検証**
    - jGrants1.0（補助金申請システム）の開発事業においては、クオリティゲートの基準を設定しTestSpecに記載した上でステークホルダーと合意形成した
    - 基準を満たしているかどうかのチェックには基本テストケース/シナリオを用いて手動でスモークテストを実施し、バグが検知された場合は開発チームに差し戻すプロセスを導入した
    - 実際には差し戻すほどのバグは検知されずテスト開始に進んだが、早い段階で基準を共有したことで、開発チームで単体テストを実施する意識付けができていたことが効果として出ていたと推察している

6. テスト自動化とTaaS

テスト実施をマンパワーで行う、いわゆる手動テストは膨大な量のテストであるほどに人や時間がかかる上、繰り返し実施するリグレッションテストには不向きであるため、テスト実施をシステムで行う、いわゆる自動テストが効果的である。一方でオンプレミスに構築する自動テストシステムは、初期コストが高い、スクリプト作成や保守にコストがかかるため、TaaS(Test as a Service)といったクラウドサービスの導入も検討の余地がある。

- ●課題
    - テスト期間中にバグを改修した場合に再テストが必要となり、実施観点では以下の2種類のテストが必要となるが、後者は影響範囲が広く不特定であることが多く、更にバグ改修の度に大量のテストを必要とするためテスト実施コストが非常に高い
        - 機能テスト（プログレッションテスト）：該当機能が正しく動作することを確認する
        - 再帰テスト（リグレッションテスト）：改修によって該当機能以外に影響を及ぼしていないことを確認する
    - 手動でテストをしている限りは現実的にテスト数を絞らざるを得ず、テスト実施者の力

量次第では影響確認が漏れる可能性が高い
    - なお、テストを自動化する上でも以下の課題が存在する
        - テスト自動化用のテストスクリプトを設計する際に、参照ドキュメントとなるテストケース/テストシナリオのフォーマットや記述の粒度がバラバラの場合に、スクリプト設計者が理解した上で設計する際にコストがかかる
        - 設計からスクリプトを実装する際に開発コストがかかる
        - プロダクトの要件変更/仕様変更に伴うシステム改修の度にスクリプトを改修するコストがかかり、またシステム改修が行われる度にステークホルダー間で情報共有が行われない場合はスクリプトが動作しない、テスト結果が異なるなどの問題が発生する
        - テストスクリプトやテストツール自体の品質保証が取れない場合、自動化によるテスト結果の信頼性が担保できず、プロダクトの品質保証を実施したとは言い難い

- ●チェックポイント
    - 3回以上同じテストを繰り返す場合はテスト自動化を積極的に検討すること
    - テスト駆動開発の場合はテスト自動化すること
    - テスト自動化の知見・経験があるテストエンジニアがいること
    - SeleniumやAppiumなどですでにテスト自動化を進めているが、その開発やスクリプト作成・保守に問題を抱えている
    - TaaSを導入する場合はテストケースからスクリプトを自動作成するサービスが提供されていること
- ●解決方針
    - テストプランニング時にテスト自動化を検討しておく
    - 自動化する上での課題やリスクを十分に理解した上で、解決案としてのTaaSの導入も検討する
- ●ベストプラクティス
    - テストケースのフォーマットはテストの自動化を見据えて、人による可読性と自動化システムへのインプットデータとしての構造化の両要件を満たす統一フォーマットとする
    - テストケースはデータとしてGitなどのリポジトリでバージョン管理する
      ※スクリプトの蓄積は避け、テストケースで一元管理することが望ましい
    - 要件変更やバグによるプロダクト改修が発生した場合はテストケースをアップデートする必要があるが、自動化システム用のスクリプトは直接アップデートせず、テストケースからジェネレータを介してスクリプトに落とし込む仕組みを用意する
      ※要件や仕様の変更はドキュメントレビューのサインオフを通じて、ステークホルダー間で情報連携できることを前提とする
    - テスト自動化が整ってきたら、JenkinsやCircleCIなどのCI/CDパイプラインの仕組みを導入し、デプロイ後に自動的にテストが実行されるように連携させる
    - テストレポートは結果情報を基に自動的に作成されるようにし、Webベースですぐにアクセスできるようにする
- **●実践検証**
    - jGrants1.0（補助金申請システム）の開発事業においては、テストプロセスを抜本的に改善したのが開発フェーズの後半であったことから、自動テストの検討は行なったが、導入までは至らなかった
    - 一方で、今後の自動化への取り組みを見据えて、テストケースはTaaSで取り込みやすいデータ構造のフォーマットを採用しつつ、人にとって可読性を損なわない工夫を施した

7. アジャイルテスターの振る舞い

組織がアジャイルを導入してもテスト方法を変えなければ、プロダクトのデリバリーサイクルでテストがボトルネックになる。テストのリードタイムを短縮するために、テスターは要件フェーズから参画し、イテレーション内での対話を通じてより早い段階からテスト設計に着手していくことが重要である。特にパフォーマンスやセキュリティなど比較的リリース間際に実施される非機能

テストにおいても、いかに早く問題を見つけ出せるかが大事である。
一方で、アジャイルテスターは、単にテスト設計・実施だけではなく、アジャイル活動を通じてチーム全体でテストを考え、自発的に実行できる文化を根付かせることが本当の役割であると言える。

- 課題
  - スプリント/イテレーションのプランニングにテスターが参画していないと、テストの要件や設計のために必要な情報をテスターにインプットするコストが発生し、デプロイサイクルを早めることができない
  - 開発メンバーがテストへの理解を持たないと、テスト駆動開発が難しい
- チェックポイント
  - オープンマインドで開発者と対話すること
  - イテレーション中に常にタスクがあること
  - 誰でもテストできるツールを導入すること
  - コードを読んだり、ちょっとした修正ができること
  - DevOps(DevSecOps)を理解し自走できること
- 解決方針
  - アジャイル開発を導入する際のロールにアジャイルテスターを取り入れる
- ベストプラクティス
  - 省内での採用ケースはまだ少ないため、今後実績を積み重ね形成していく
- **実践検証**
  - jGrants1.0（補助金申請システム）の開発事業においては、開発チームのTDD（テスト駆動開発）を促進するためにアジャイルテスターの採用を試みたが、開発チームからの難色により頓挫している

8. 参考情報

1) 共通して活用すべきフレームワーク

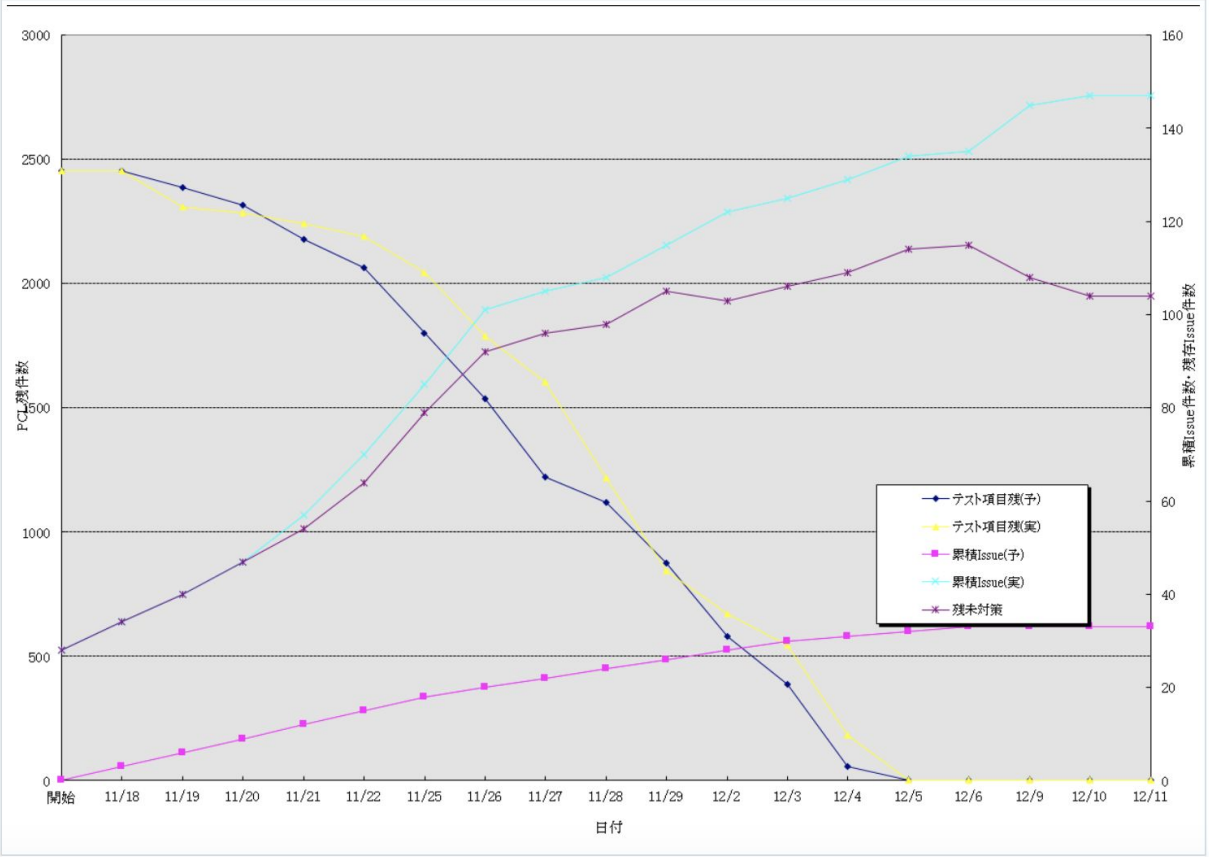
- サインオフプロセス
- クオリティゲート
- トリアージプロセス
- DevOps/DevSecOps

2) 共通して活用すべきITツール

- クラウド型自動テスト(TaaS: Test as a Service)
  https://www.intec.co.jp/company/itj/itj13/contents/itj13_30-37.pdf
- セキュリティチェックツール
  - OWASP ZAPなどが活用できる
- バージョン管理ツール
  - GitHub/BitBucketなどが活用できる
- バグトラッキングシステム(BTS: Bug Tracking System)
  - チケット管理ツール(JIRAなど)をベースにしたバグ管理専用のボード
- 信頼度成長曲線(SRGC: Software Reliability Growth Curve)
  https://itmanabi.com/reliability-growth-curve/
  - SRATS/SRATS2010などが活用できる
  - 以下にSRGCの例を記載する

issue状況グラフ

12/10時点



動的に変化する予測グラフを活用すれば、いつからバグ検知数が飽和していくのかを事前に把握することができるため、テスト結果を知るよりも前にリリース日を確定するための情報材料として利用することができる　※必ずしも正確ではないため、判断する説得理由の材料が乏しい場合の拠り所に留める

- CI/CDパイプラインツール
    - Jenkins/CircleCIなどが活用できる

3) まとめるべきドキュメント

- テスト計画書
    - テストゴールやテスト要件、テストスコープ、テスト方法やテスト環境、体制やスケジュール、リスクなどの情報を記載
- テスト設計書
    - テストスコープ詳細やテスト詳細などの情報を記載
- テストケース/テストシナリオ
    - テスト設計書に別添で内包
    - テスト実施の入力条件やパターン、期待結果、ステータスを記載
- テストデイリーレポート
    - テスト実施期間中にテストの進捗や結果、検知したバグの情報を記載
- 振り返りドキュメント
    - テスト活動全般におけるKPT&Aをメンバー間で議論、共有
- テスト完了レポート
    - テスト実施後にテスト結果やバグトリアージ結果を記載
    - 必要に応じて、振り返りで議論した今後の施策を記載

4) 参照すべき政府・経済産業省のルール

- デジタル・ガバメント推進標準ガイドライン解説書（第7章）
  https://cio.go.jp/sites/default/files/uploads/documents/kaisetusyo_3-7.pdf
- 高信頼化ソフトウェアのための開発手法ガイドブック
  https://www.ipa.go.jp/files/000005144.pdf

■デプロイ

--- To Be Announced ---

3.3 サービスを育てる

3.3.1 保守・運用

--- To Be Announced ---

3.3.2 カスタマーサポート

0. カスタマーサポートとは

カスタマーサポートは利用者の生の声が集まる貴重なチャンネルを運営するものであり、また利用者のサービス満足度に直結する営みである。プロジェクト開始段階では開発物が重要視され、カスタマーサポートの検討は後工程とされがちであるが、DevOpsのライフサイクルにカスタマーサポートの自動化やフィードバックを取り込み改善を図るプロセスを早期に盛り込んでおくことで顧客体験の向上を図るだけでなく、運用の効率化、政策評価に有効なフィードバックの取得につなげることができる。

1. カスタマーサポートにおけるアクター別の課題

1-1. 利用者（申請を行う人）

例えば行政が運営する電子申請サイトで申請を行う利用者にとって、目的はそのシステムで実行できるタスク（制度の検索や申請処理など）が最小限の労力で完結することであり、システムに長時間滞在すること自体苦痛である。その観点から事業者の立場にたったサポートを設計することが重要となる。

- 必要な情報がどこにあるかわからない
  利用者がこれから行うシステム操作について、事前にじっくりマニュアルを読み込みシステム操作を理解した上で操作を開始するというシナリオは稀である。基本的には操作を始めてみて、つまづいたところでサポートを求める。その際、自分が操作しているページ上に必要な情報がないとその時点で利用者は不便を感じ離脱のポイントとなる恐れがある。また、なんとかTopページなどからマニュアルを見つけたとしてもそのマニュアルが数十、数百ページに及ぶ細かなマニュアルであり、目次を見て構成を理解するところから必要であった場合それも不便・負担を感じるポイントとなり得る。

- 今必要な疑問についてのみ解決策が知りたいが見つけられない
  利用者は今直面している問題に対する回答のみを労力をかけずに得たい。詳細な操作マニュアルの中から解決策を探すのは利用者にとって大きな負担である。

- タスクを進めるのに待ち時間を発生させたくない
  利用者は最低限の時間でタスクを完結させたい。そのためには問題に対する回答もスピーディに得ることが必要である。例えばコールセンターを備えていた場合、電話でオペレータに問い合わせ適切な回答を得るのは1つの解決策であるが、コールセンターの営業時間外であったり、あるいは申請締切日間際でコールセンターが込み合っていて電話がつながらない時も利用

者が負担を感じるポイントである。また電話がつながって、自分の直面する問題を伝えることができたとして、そこで即回答が得られず折り返しの電話待ちとなるのも利用者としては負担であろう。外部要因による待ち時間の発生が回避できるよう複数のサポートを用意することが重要である。

1-2. サービス運営者（行政事務を行う人）

サービス運営を行う立場からすると、利用者からの細かな問いに都度対応していては事務コストが高い。利用者の疑問解決にはなるべく効率的な手法で解決しつつ、利用者が操作につまづいたポイントやこういう機能が欲しいといったニーズを抽出し、次の制度設計やシステム改修に有効なフィードバックを得ることができるよう設計することが重要となる。

- 利用者がどこで困っているかわからない
  システムの操作を把握しているサービス運営者からすると、初見でシステム操作する利用者、あるいはITリテラシーの異なる利用者（普段PCの操作を行わない、スマホをもっていない、等）がどういうポイントでつまづくのかわからない。あるいはその利用者のおかれた環境によってつまづくポイントは異なり、サービス運営者側で想像が及ぶ範囲は広くない。リリースまでの間にモニターテストやデモ環境の公開によりシステムの操作を行った人からフィードバックを得てFAQを用意しておくことはできるが、本番の申請操作や実際のフローを体験してみて初めて疑問に思う点や不便に思うポイントは多いため、完璧なFAQを事前に用意することはできない。

- 誰にエスカレーションすればいいかわからない
  問い合わせにはあらゆる観点のものが含まれる。システム操作の質問、制度に対する質問、手続きの流れに対する質問、トラブルおよびその解決策、苦情、改善ニーズ、あるいは制度そのものへのご意見など。来た質問に対しヘルプデスクの過去知見で回答ができない場合、誰に回答を求めるか、どのように合意形成を取った後回答を返すかから考えていては回答が返せるまでに時間がかかってしまう。

- 対応ノウハウの蓄積・共有ができない
  サービス運営者がメールや電話で個別にくる問い合わせを都度返すのみでは対応ノウハウは蓄積されないし、他のメンバーと共有もできない。第一歩としてExcelで管理簿をつけることがコストもかからずすぐ始められる対策であるが、件数が多くなってくると共同編集による問題が発生したり、また担当者毎の記入ブレや漏れにより検索性が悪くなり、管理すること自体にコストがかかってしまう。

- 次の開発にフィードバックできていない
  情報を蓄積し共有するだけでは次の開発で取り組むべき優先度の高い課題は抽出できない。問い合わせで声の大きかった人の意見、とくに行政では影響力のあるステークホルダーの意見を優先的に取り上げてしまうことが多いが、それが本当に多数の利用者にとって価値がありそのプロダクトが目指すゴールを満たす上で優先度の高いものであるとは限らない。問い合わせから利用者が本当に解決したい課題を見つけるためには、表面的な声だけでなくその問い合わせに至った背景まで理解できることが重要である。

2. 課題の解決方針（ベストプラクティス）

2-1. 利用者（申請を行う人）むけ

- チャットボットの提供
  必要な情報を見つける、今必要な解決策だけほしい、回答を待ちたくない、これら全ての対策の1つとしてチャットボットは有効である。また、チャットボットをSaaSとして提供するサービスを活用すれば、開発への影響は最小限で導入することができる。（例：補助金申請システム（以

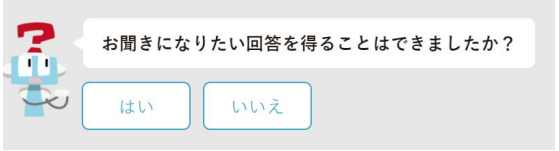
降jGrants）で導入するai-FAQの場合、契約者毎に発行されるスクリプトタグをチャットボットを出したいページに追加するのみ）
ただし、利用者の疑問にヒットするようなチャットボットのFAQを用意・メンテナンスしていくのは難しい。ここではチャットボットを設計・運営する上で実施した事例を紹介する。

- 実際の利用者の声をひろう
  システムを構築している立場からでは利用者の目線にたったFAQの作成は困難である。

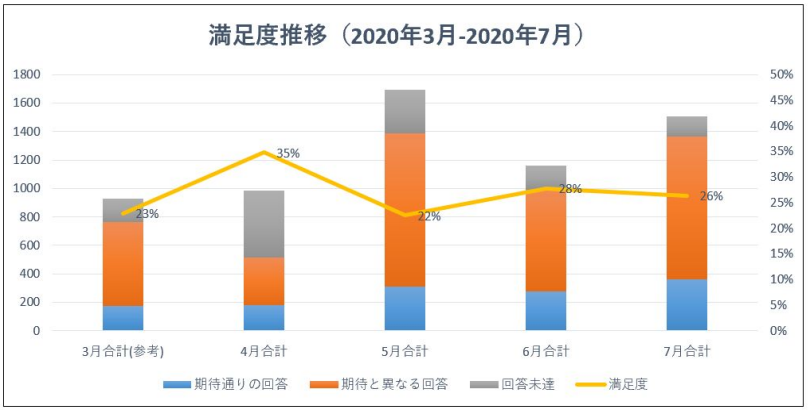
- システムスコープ外のFAQも含める
  事前に行ったモニターテストやユーザテスト、説明会をする上で利用者が実際操作につまったり、質問が多かったもの、あるいはマニュアルだけでは読み取れないものを体系化し、FAQに整理をするべきである。なるべく実際の利用者が置かれているのと同じ状況・環境（初めて該当システムを触る、マニュアルは読んでいない、PCの操作にはあまり慣れていない、補助金の申請の流れは把握できていない、会社のPCは遅い、IEしか使えない、等）で操作をした人の声を集めることが重要である。
  FAQ作成にあたり、連携する他システムに関する問い合わせは他システムのサービスデスクなどの窓口に誘導することが正しい情報を提供する上では正しいことであるが、利用者からみるとたらいまわしにされたように感じる。例えばjGrantsではログインに法人共通認証基盤（以降、gBizID）を活用しているが、チャットボットアクセスの10%程がログインに関するFAQ（アカウントの取得方法、ログインできない、パスワードがわからない、等）である。これらを全てgBizIDのサービスデスクに誘導することは利用者の満足度を下げることにつながるため、代表的なFAQの回答は追加をしている。利用者が複数のサイトを行き来することなく疑問解決につながることが重要である。

- 適切な画面にチャットボットを配置する
  必要な情報がどこにあるかわからない、今必要な解決策だけほしいという課題の対策となり得るが、チャットボットは一律同じものを全画面から呼び出せるようにするのではなく、そのシーンに適したものを配置することが好ましい。例えば申請画面上では質問の候補としてあらかじめ申請時によくアクセスされるFAQをチャットボット画面上に配置しておくことなどで、利用者はすぐに目的の回答を得ることができる。

- 利用状況分析を行い、FAQを改善する
  チャットボットは日々の利用状況の分析とFAQのブラッシュアップを行い利用者の満足度を上げるための改善サイクルを回すことが重要である。例えばjGrantsでは以下のような形で満足度を集計し、推移や傾向を分析している。また、FAQ毎の回答の満足度を分析し、どういう回答が期待されるものであったか仮説をたてた上でFAQの見直しを毎月行っている。
  - 満足度の集計
    ai-FAQではチャットボットの回答の後、以下のようなアンケートをとっており、その結果を管理画面上で集計できる。



    ここで①「はい」を選んだ人を期待通りの回答、②「いいえ」を選んだ人を期待と異なる回答、答えなかった人を「回答未達」として集計し、　　満足度＝①÷（①＋②）として算出。それを月ごとに集計し推移を図っている。



この推移をみることで後段で述べるFAQの更新などの改善に効果があったか傾向を見ている。またあわせて利用数の推移もみることで活用が進んでいるのか、なにかイベントがあって増えているのかなどもあわせて分析している。

■ FAQ単位の分析

FAQの単位で満足度の低いものをピックアップし、改善を図っている。まずFAQの表示回数により3つのグループに分類し、

・表示回数の多いグループ　すべて分析対象

・表示回数の少ないグループ　すべて分析対象外

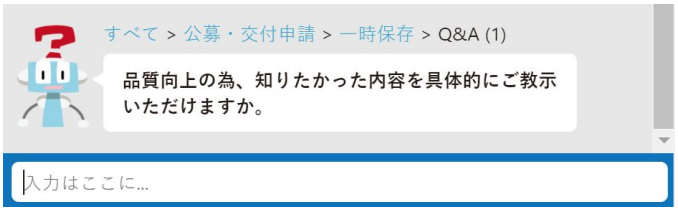
とし、中間のグループは、上述の「いいえ」を選んだ人あるいは回答未達の割合が高いものをピックアップし分析対象とする。これにより分析対象のQAを110件中36件に絞っている。これは、あまり表示されていないFAQや、表示回数もさほど高くなく満足度もさほど低くないものを除外することで、改善の効果が大きいFAQを絞り込むためである。

また、上記の36件について質問の分類毎に「いいえ」を選んだ人あるいは回答未達の割合を算出し、特にどの分類で満足度が低いのか分析している。

単位：質問件数

質問の分類	6月				7月			
	回答なし	期待と異なる回答	計	割合	回答なし	期待と異なる回答	計	割合
①	32	124	156	27%	46	223	269	29%
②	15	155	170	29%	18	231	249	27%
③	6	25	31	5%	3	70	73	8%
④	26	71	97	17%	24	69	93	10%
⑤	11	37	48	8%	8	97	105	11%
⑥	23	60	83	14%	44	100	144	15%
合計	113	472	585	100%	143	790	933	100%

ここで、とくに満足度の低い分類のFAQを中心に、実際利用者がどのような回答を求めていたか仮説をたてる。仮説にあたってはai-FAQで「いいえ」を選んだ人に対し以下のようなアンケートをとっていることからここに書かれた内容を参考にしている。また、FAQを選択する際にキーワードで検索することができるが、その情報も参考にしている。



たとえば、5月はチャットボット利用数が多いにも関わらず満足度が低い。分析の結果「コロナ」といったキーワードや、コロナに関する給付金がここから申し込めるかといった内容が多いことがわかったため、コロナ対策制度へのリンクやjGrantsで取り扱っている補助金などのQAを掲載したところFAQアクセスランキングトップとなり翌月は満足度がやや上昇した。

- コールセンターの提供
  必要な情報を見つける、今必要な解決策だけほしい、という課題に対し、一定数以上の利用者が想定される場合にはコールセンターの運営が有効である。以下に利用者の課題を解決する上で考慮すべき点をあげる。

    - インハウスかアウトソースか
      自組織の従業員が電話応答するか、業務を外部委託するかはまず最初に検討すべき点である。自組織にサービス運営部署をかかえているような場合、まずはそこでオペレータを配置することが考えられる。組織内で業務知識を共有できることから高いサービスを提供できる一方、繁閑の差への対応が難しかったり、サービス提供時間とオペレータの勤務時間が合わないことなどで、利用者としてはサービスを受けたいタイミングでオペレータにつながらない状況が発生する。また単純にサービス規模に見合ったリソースを自組織のみで確保できない場合もある。そのような場合アウトソースは1つの解決策である。

    - チャットボット、メールとの併用
      コールセンターはコストが高いため、なるべく自動応答で解決できる課題はチャットボットに誘導する、急ぎでないものはお問い合わせフォームからメールでもらう、などによりコールセンターへの問い合わせを減らすことが望ましい。チャットボットでまず質問をはじめ、自動応答の回答で解決に至らなかった場合に限りコールセンターに接続するといった実装もよく見られる。

    - ツールの活用
      コールセンターにて利用者の課題を迅速に解決するため、活用を検討すべきツールを以下にあげる。
        - コールセンター管理ツール
          利用者からの電話を受けた際、着信数や待ち呼数などの状況がダッシュボードで管理できたり、問い合わせ履歴やナレッジの検索、問い合わせ内容の入力や録音など、コールセンターに必要な機能を備えたツール。このようなツールの活用によりコールセンター側の受付を効率化することで、利用者へ正しい情報を素早く提供することができる。
        - 音声自動応答システム（IVR）
          利用者が電話をかけた際、音声ガイダンスに従い問い合わせの分類などを選択させることで自動的に音声案内をしたり着信先を振り分ける仕組み。これを音声ガイダンスではなくWebブラウザまたはアプリ上のメニューとして提供し選択結果により電話やチャットといったチャンネルにふりわける、「ビジュアルIVR」と呼ばれるサービスもある。利用者としては自動振り分けにより正しい担当につながるため、たらい回しにされたり待ち時間の削減につながる。Biztelなどクラウド型のコールセンター機能を提供するサービスにはオプションとして含まれている場合がある。

- コンテンツの提供
  利用者が課題解決するために複数のサポートを用意し、利用者が状況や理解度によって適したものを活用し解決できることが重要である。チャットボットやサービスデスクの他にも以下のようなサポートが効果的である。
    - クイックマニュアル
      利用者がシステム利用にあたり最初にやることや必ず行う操作を簡潔にまとめたもの。画面キャプチャーや画像を活用しわかりやすさを重視すること。
    - 操作動画
      実際の操作をナレーションや字幕入りで動画にしたものを公開したもの。ログイン、申請、など操作の単位にわけ3分程度にまとめるのが見やすい。
    - FAQ

よく問われるFAQはサイト上にページを用意する、あるいは一覧をダウンロードできるようにしておくのも有効である。

2-2. サービス運営者（行政事務を行う人）むけ

- 各種指標を用いた利用者行動の分析
  利用者がどこでつまづいているかわからないことへの対策として、利用者が一連の操作を行うための導線と、その要所毎に定量的に取得できる経過指標をあらかじめ分析の上決定し、運用開始後どのような契機でどのような分析、それに基づきどのような対策を行うのか設計フェーズで検討しておくことが重要である。経過指標は取得のために開発が必要なもの、GoogleAnalyticsなどのツールが必要なものなどがあるため、開発の前に洗い出しておき、設計に含めておく必要がある。
  経過指標は、例えばjGrantsにおいて「補助金を見つける～申請をする」という導線においては以下のようなものを想定している。
    - 流入数・直帰数（アクセスされた数、サイト内の他ページに移らず離脱した数）
    - ログイン数
    - 申請開始数（公募・交付申請を開始した数）
    - 申請提出数（公募・交付申請が提出された数）
    - 差戻し数（公募・交付申請が差し戻された数）
    - 申請完了数（公募・交付申請が完了した数）

  申請開始数に対し申請提出数が著しく減っていた場合、入力が煩雑であったり途中で紙申請に切り替えてしまった等、申請入力になにか課題があり離脱があったと仮説が立てられる。その場合特に離脱の多い補助金の申請項目を調べる等、これを足掛かりに課題を深堀し、UI/UX改善につなげたり、そのシーンに対するチャットボットFAQやサポートを充実させるなどの対策案につなげることができる。

- ツールによるチケット管理
  コールセンターの知見や過去ノウハウだけで解決できない課題や、保守対応でデータ修正が必要となるような事象が発生した際プロジェクト内でのエスカレーションが必要となる。このようなタスクのステータス管理およびプロジェクト内のワークフローの自動化にはチケット管理ツールなどを活用し効率化することが重要である。以下活用事例を紹介する。

    - Redmine
      カンバン方式でチケット管理が可能。タスクが発生したらチケットを発券し、事前に定めたルールに従い担当者を指定することで担当に通知（メールなど）ができる。また、ステイタスを例えば
        - Backlog　チケットを発券した状態
        - Current　処理を始めた状態
        - NeedsReview　内容により必要な承認者に承認を得る
        - Finished　処理完了

      としカンバンモードでステイタス毎にチケットを並べて見ることができるため、状況を俯瞰して把握しやすい。
      また個々のチケットに調整で活用した資料や経緯をコメントできるため、後から検索ができノウハウの共有や蓄積ができる。あるいは
        - wiki　共有のノウハウなどを共同編集しブラウザ上で参照できる
        - ファイル　フォルダを作成しファイルをアップロード・保管できる

      などの機能を活用し、各チケットであつまったノウハウをwikiに編集したり資料を分類してアップしておくことなどでよりノウハウの共有がスムーズとなる。
      なお同様のツールにBacklogという製品もある。

- DevSecOpsへの組み込み
  開発者（Dev）と運用者（Ops）が協力しあってユーザーに迅速かつ継続的にプロダクトやサービスの提供を行うことをDevOpsと呼び、近年その実装の開始時点からセキュリティ（Sec）も組み

込んだモデルとしてDevSecOpsと呼ばれるようになった。一般的に以下のようなプロセスを回すモデルとなっている。

1. 計画（PLAN）
2. ビルド（BUILD）
3. 継続的インテグレーション（CONTINUOUS INTEGRATION）
4. デプロイ（DEPLOY）
5. オペレーション（OPERATE）
6. 継続的フィードバック（CONTINUOUS FEEDBACK）

この中で6の継続的フィードバックが、カスタマーサポートでユーザから得たフィードバックを次の開発につなげる営みである。フィードバックに基づき改善を行うサイクルを確立することで、高いアジリティを維持し顧客の要求に素早く対応することができる。
例えばjGrantsでは現在次期バージョンの開発を進めているが、現行バージョンの開発・運用をする中で集めた現行システムに対する課題や業務そのものに対する課題等をIssueリストとして整理し、その中から解決すべき優先度の高いIssueを抽出し設計・検討に反映している。これらの管理を統一的なツールで一元化し、Issueの書き込みや分析が効率的にできると今後DevSecOpsのサイクルをよりスピーディにまわすことができる。

- チャットボットの活用

「利用者（申請を行う人）むけ」の章でチャットボットについて触れたが、チャットボットはサービス運営者（行政事務を行う人）側のメリットもある。

- 事務コストの削減
  利用者の章で述べたとおりチャットボットを正しく運用し回答率を上げるための改善を続ける必要はあるが、チャットボットで解決できればサービス運営者向けの個別問い合わせ件数を抑えることができる。
- 利用者からのフィードバックを得ることができる
  チャットボットも利用者との重要なタッチポイントの1つである。jGrantsで運用するチャットボットでも課題解決に至らなかった利用者にアンケート欄を表示しているが、そこには潜在的なニーズなどが含まれる場合がある。例えば「こういう操作はできるか？」という入力があったとき、潜在的にその操作に対するニーズがある可能性があるため、同じような声は多いのか、その操作ができることでどのような価値を生み出すのか、深堀をすることで今後のUI/UX改善につなげることができる。

3. 参考情報

1) 共通して活用すべきITツール

- チャットボット
  ai-faq
- コールセンター
  Biztel
  PureCloud
- チケット管理
  Redmine
  Backlog
- コミュニケーション・ワークフロー
  Slack
- アクセス解析
  Google Analytics
  UserInsight

3.3.3 データ分析・マーケティング

サービスのリリースをもって、サービス開発のタスクが完了するわけではない。ユーザーの満足度を更に向上させるため、データ分析を通じてサービスの利用状況や改善機会を把握し、継続的にサービスの改善を図っていく必要がある。具体的には、①デザイン（UX向上のための機能設計）、②実装（プロトタイプの構築）、③評価（データ分析を通じた改善点の把握）のサイクルを継続的かつ反復的に回していくことで、より良いサービスを作り上げていくことが重要である。
本稿では、データ分析を通じたデジタルサービス（Webシステム）の改善という観点から、発生しうる主な課題とその対応案について記載する。

1．データ分析における課題

データ分析のフェーズにおいては、主に以下のような課題が発生しうる。

- 改善につながるKPIを設計できていない（1-1）
- ユーザー属性に応じた課題把握ができていない（1-2）
- ユーザーの本音（要望や不満）が深掘り出来ていない（1-3）
- 改善のPDCAサイクルを上手く回せていない（1-4）

1-1. 改善につながるKPIを設計できていない

- そもそもKPIを定義しておらず、担当者のKKD（勘・経験・度胸）によって、場当たり的にサービス改善を図ろうとしている／もしくは何もしていない。
- 様々なKPIを定義し、ダッシュボードを構築したが、各指標の位置づけや関係性が明確になっておらず、結局は使われないものになってしまう。
- UXの改善を目的とした一貫性のあるKPIとなっておらず、”どの施策によって、どこにどれだけの効果があったのか”が検証できない。

1-2. ユーザー属性に応じた課題把握ができていない

- 全てのユーザーをひとまとめに考えてしまっており、KPIを見ても、具体的にどのようなユーザーが抱えている問題なのか切り分けができない。結果的に、それぞれの特性や利用シーンに応じたきめ細かなサービス改善が図れない。
- 公共事業である以上、どのようなユーザーに対しても公平で使いやすいサービスを提供する必要があるが、特定のセグメント（PC弱者など）についての配慮が置き去りになってしまっている。

1-3. ユーザーの本音（要望や不満）が深掘りできていない

- KPIのトラッキングを通じて、サービスがどのように利用されているのかは把握したが、どのようにサービスを改善すればいいかはよくわからない。また、データ化されていない利用者の思いについては把握ができない。
- 結果、省庁側の思い込みで必要と思われる機能を実装したが、結果的にユーザーに利用されず、無駄な工数となってしまう。

1-4. 改善のPDCAサイクルを上手く回せていない

- サービスの新機能やUIを変更して全面的にユーザーに展開したが、評判が悪く、結果として全ユーザーの満足度を下げてしまう。（サービス前の仮説検証ができていない。）
- また、機能変更の前後でKPIを比較しているが、タイミングがずれていることで前提条件の差異（繁忙期など）が発生し、どこまでが新機能による影響要因なのかの切り分けができない。

2. データ分析の進め方

データ分析のフェーズは、上述の課題に対応するため、大きく以下のような点に留意して進める必要がある。

- ユーザーの行動を起点にKPIを組み立てる（2-1）
- ユーザーをセグメント化し、行動を分析する（2-2）
- 定量データと定性データを組み合わせる（2-3）
- 実験的な仮説検証サイクルを繰り返す（2-4）

2-1. ユーザーの行動を起点にKPIを組み立てる

- デジタルサービスの改善に当たっては、最終的なゴールを設定し、その実現に向けた指標をモレれやダブりがない形でKPIとしてブレイクダウンすることが重要である。
- 本章では、「電子申請率の向上」（＝UX改善による申請者の満足度向上）というゴールを設定し、その実現に向けたKPI群を定義する。
- この際、ユーザーの行動プロセスをEnd-to-Endのステップに分解し、どのステップにどのような問題が発生しているのかを示せるKPIを定義する必要がある。
- 本章では、こうした「ファネル分析」に向けて、グロースハックのフレームワークであるAARRR（アー）をベースとしたKPI設定について記述する。

AARRRのフレームワーク

- 本フレームワークでは、ユーザーの行動を以下5つのステップとして定義し、それぞれに対してKPIを設定していく。

ステップ	説明
Acquisition（獲得）	● 様々なチャネルからサイトにアクセスする
Activation（活性化）	● 初回のログインを実施する
Retention（継続）	● 定期的にサイトにログインするようになる
Resolve（解決）	● 対象の手続きを完了する（=問題解決）
Referral（紹介）	● サービスを口コミで広げる

※本フレームワークでは、本来はRevenue（収益）が使われているが、営利活　動を目的としたサービス ではないため、Resolve（問題解決）と変更する。

- 各ステップごとにKPIを設定し、トラッキングすることで、どの段階に問題が生じているのかを把握することができ、PDCAを回すことによるサービス改善につなげることができる。

ステップ	KPI（例）	改善施策（例）
Acquisition （獲得）	● 1日あたりのWebサイト訪問者数 ● 流入元のチャネル分布	● 広報強化 ● SEO
Activation （活性化）	● ランディングページ（ログイン画面）の平均滞在時間 ● 1日あたりのログイン数（新規）	● ランディングページ改修 ● GビズID活用

Retention （継続）	● 1日あたりのログイン数（既存） ● 各ページの平均滞在時間 ● 各機能の利用回数・時間	● レイアウト、機能改修
Resolve （解決）	● 1日あたりの申請数 ● コンバージョン率 （例：申請完了者数／ユーザー数） ● 申請エラー率（差戻しの割合） ● 審査リードタイム	● レイアウト、機能改修 ● 制度・運用見直し
Referral （紹介）	● SNSへの投稿コメント	● 機能改修 ● 制度・運用見直し

    - これらのKPIをダッシュボード化し、業務のルーチンとして定期的にチェックすることで、サービスの現状について把握することができる。

- サイト及びプロセス全体に関わるKPIについては上述の通りであるが、新機能のリリース時などは、より細かい「機能」の単位でKPIをトラッキングすることも重要である。機能レベルのKPIとしては、大きく以下のような観点が考えられる。

    - ユーザー全体に対する機能の利用率
    - 特定期間における機能の利用頻度

- クリック数などのイベントデータを用いてKPIのダッシュボードを構築し、KPIのトラッキングを通じた機能改善を図っていくこと。また、機能の利用有無でユーザーをグルーピングし、上述のKPIに対して有意な差異が確認されるかを確認することも有効である。

2-2. ユーザーをセグメント化し、行動を分析する

- サービスの利用主体であるユーザーは誰もが同じ特性を持っているわけではない。そのため、ユーザーをセグメント化し、ユーザーのタイプごとにサービスが効果的なのか、改善の余地がないのかを探っていく必要がある。
- ユーザーをセグメント化するための分類軸としては、以下のようなものが考えられる。システムに登録されているデータをもとに、軸として利用可能なものを洗い出すこと。

分類軸	具体例
利用開始日	● 初回ログイン日・月
属性	● ユーザーの年齢や住所、事業体の規模
行動特性	● デバイスやアクセス頻度

- 属性情報については、ユーザ登録や申請のタイミングでユーザーに入力してもらうことになるが、要件定義／システム設計タイミングから、分析という用途を意識しておくことが重要である。（例：テキスト情報ではなく、リスト化してユーザーに選択させる、等）
- ユーザーのセグメントを定義した後、2-1で設定したKPIに対して、セグメント間の差異を確認する。その結果、サービスの利用率の低いセグメントに対しては、セグメントの特性を考慮した追加機能を提供することでUX改善を図る、といった打ち手を検討する。

- ファネル分析（2-1）やユーザーセグメント分析（2-2）には、Google Analytics等のWeb解析ツールを用いると効果的であるが、個人情報の取り扱いには十分に配慮する必要がある。ツールの利用にあたっては、サービスの利用規約／サービスポリシーにWeb解析ツールを使用している旨を明記し、ユーザーの事前同意を得ておくこと。

2-3. 定量データと定性データを組み合わせる

- 分析に用いるデータは、大きく「定量データ」と「定性データ」に大別される。サービス改善にあたっては、それぞれの利点や欠点を理解し、組み合わせて活用することが効果的である。

種別	説明	用途・特徴
定量データ	● システムから収集できるサービスのログ情報	● ユーザーグループごとの特性や差異の把握に有効 ● ユーザーの行動理由に関する示唆は限定的であり、結果の解釈が必要
定性データ	● インタビュー等を通じて収集する（主に）テキスト情報	● ユーザーの要望や不満などに対する深い把握に有効 ● データ収集に手間がかかり、必ずしも正確であるとは限らない（バイアスあり）

- 定量データと定性データを用いたサービス改善については、以下のようなサイクルを繰り返していくことが効果的である。
    - ダッシュボードなどを用いて「定量データ」（KPI）を把握し、サービスの利用状況や改善領域を特定する
    - 特定した改善領域について、ユーザーヒアリングなどを通じて取得した「定性データ」を活用して、その原因や対応案についての仮説を立案する
    - 課題を解決するための機能を実装し（プロトタイピング）、実験的にリリースすることで効果検証を行う

- 定性データの調査手法としては、以下のような手法が代表的なものとして挙げられる。

調査手法	具体例
インタビュー	● 1対1のミーティング
フォーカスグループ	● 複数人でのディスカッション
ユーザビリティテスト	● プロトタイプを活用したデモ
SNS分析	● Twitter等への投稿コメント確認

- 上記の手法を通じて把握したユーザーの要望については、実装すべき機能にまで落とし込んだ上で、インパクトやコストといった観点で優先度評価をし、次のスプリントでの仮説検証を進めていく。

2-4. 実験的な仮説検証サイクルを繰り返す

- 実施した変更が既存バージョンと比較してKPIの改善につながるのかという仮説検証については、「A/Bテスト」が有効である。A/Bテストの実施は、大きく以下のステップにて実施する。

    - 変更の内容と対象とするユーザーセグメントを決定する
    - 変更内容を実験的に適用し、対象ユーザーに開放する
    - KPIを測定し、変更内容が改善につながったか確認する

- A/Bテストは、様々な要因による影響を最小限にするために、原則として1つの変更に対して実施するが、複数の変更を同時にチェックするためには「多変量テスト」を用いて、一度に様々なバリエーションのサイトをテストする。
- また、実験に際しては、（上述の通り）比較対象が同じユーザーセグメントに属していることが必要不可欠である。差異が確認された場合、それがユーザー特性に起因するものでなく、サービス側の変更に起因するものであると切り分けられるように、ユーザーのセグメントは統一しておく必要がある。
- A/Bテストの実施に際しては、一つのKPIに注力して実験のサイクルを回していくことが効果的である。複数のKPIを用いた場合、1つのKPIは改善しているが、その他が悪化している場合どうするのか、といった問題が生じてしまう。よりスムーズな意思決定のためには、目的に応じてKPIを絞り込むことが重要である。
- 前述のGoogle AnalyticsなどのWeb解析ツールを活用することで、より簡易にA/Bテストを実施することができる。単純なKPIの比較だけでなく、統計的な分析が必要な場合、各種の分析ツール（Alteryx等）を組み合わせることで、より高度な統計分析が可能である。しかし、高度な分析ツールやテクニックの前に、まずは基本的なKPIを使いこなせるようになることが最優先である。

3. 参考情報

1) 共通して活用すべきフレームワーク

- AARRR
  https://www.innovation.co.jp/urumo/aarrr/

2) 共通して活用すべきITツール

- Google Analytics（Webアクセス解析）
  https://analytics.google.com/analytics/web/provision/#/provision

- Tableau（ダッシュボード全般、Google Analytics連携可能）
  https://www.tableau.com/ja-jp

- Alteryx（データ集計・分析など、Tableau連携可能）
  https://www.alteryx.com/ja

4. その他全プロセスで共通して考慮すべき事項

セキュリティ

概要

セキュリティは手段であるということを忘れてはいけない。守る対象があり、それを守るための手段の総称が、サービス開発におけるセキュリティである。
実際の参照すべきガイドラインなどは各章でも提示されるが、ここでは全体概要や考え方について簡単に記載したい。
特に、周辺の環境を条件にセキュリティ要件を緩和するようなことはあってはならない。具体的には、『境界型防御の内側なので条件Aは緩和してよい』等が典型例となる。同様に、『利用者が内部の人間なのでアクセス制御、権限制御機構を導入しない』等も決して許容してはいけない。認証のないシステムは理論上はありうるが、現実的にはほぼあり得ない。この場合、認証機構は決して自前で作成せず、可能な限り既存の評価済みの設計や実装を利用する必要がある。何に準拠したかわからない認証機構はむしろ危険だと認識するべきである。暗号アルゴリズムの選定に関しても、独自のアプローチは避け、国内でいえばCRYPTRECの評価を元にNISTの基準をフォローアップとして参考にすべきだろう。(CRYPTRECガイドは即応性に欠ける為)
標準ルールがあればそれに則るべきだが、アカウントのライフサイクルマネジメントに関しては、設計段階から検討する必要がある。

原則

重要なのは、***サービスデザインの段階でセキュリティの検討を済ませておくことである。***「Secure by Default」である必要があり、セキュリティ機構のないシステムは本来存在しない。
「シフトレフト」等ともいわれるように、初期から検討されたセキュリティは強固であり、安全性、メンテナンス性、持続性の向上に寄与する。一方、後付けのセキュリティは、それがシステムであろうが人的なオペレーションであろうが等しく脆弱であり、コストが高い。(脆弱さとコストの高さは往々にして相関関係にある)
DevSecOpsの概念は重要であり、官公庁の調達ではなかなか実現が難しい側面もあるが、デジタル化推進マネージャーの存在はそれを一定量解決する手段足りうるだろう。また、可能であれば、第三者のレビューや診断などを定期的に受けるべきである。
システムのアーキテクチャーには王道が存在する。予算都合などでそれを歪めることは多々あると思われるが、その際にセキュリティの側面におけるインパクトを「必ず」検討すべきである。
セキュリティにおいては、決して属人的な概念で設計や設定をしてはならない。必ずロールベースやアトリビュートベースでの論理設計をし、システム設計、環境設定を行った上で運用を行う必要がある。
e.g.) データベースの適切なアカウントと権限の設定がされているか等
(ロールベースの最小限の権限設定が行われている必要がある)

参考資料

- 体系的に学ぶ 安全なWebアプリケーションの作り方 第2版 脆弱性が生まれる原理と対策の実践
- DevOps 技術: セキュリティのシフトレフト
- CRYPTREC暗号リスト(電子政府推奨暗号リスト)

- NIST SP 800-63 Digital Identity Guidelines (NIST Special Publication 800-63-3 OpenID Foundation Japanによる日本語翻訳版)

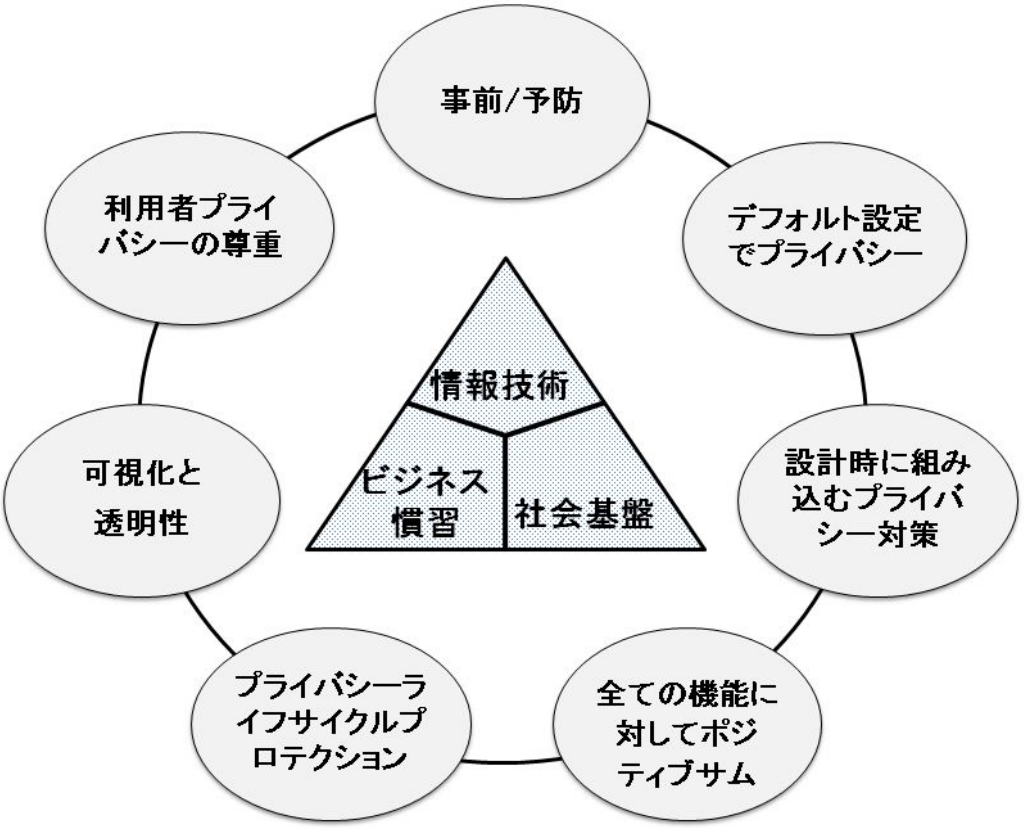
プライバシー

概要

デジタル化されたシステムにおいて、プライバシーの取り扱いは非常に重要である。現在のインターネットのような情報伝達システムを人類が保有したのはつい近年のことであり、取り扱いや社会受容の概念はまだまだ未成熟な状態であるため、慎重に取り扱う必要がある。
プライバシーとプライバシー保護は別の話であり、前者はサービス設計の根幹に関わる場合がある。また、後者はサービス設計の段階で保護手法を、セキュリティと織り交ぜながら検討していく必要がある。なお、ここからわかるように**プライバシーはセキュリティに分類されるものではなく**、プライバシー**保護**の為にセキュリティが重要なのであり、ここではセキュリティは手段、プライバシー保護は目的である。なお、執筆者の経験上、プライバシーに関する取り扱いは非常に難しく、高い専門性が求められるため、一定のアドバイザーに足る専門家リソースをプールしておく必要がある。一部の政府CIO補佐官など、相談できる先を常にキープしておき、PIAをするとまではいわずとも、必ずプロジェクトごとにアドバイスを求めるべきである。(これはセキュリティにおいても同様だが、より人材難である為)
e.g.) 我が国の令和2年改正個人情報保護法(*)やEU GDPRやアメリカCCPA、APEC CBPR等、カバーすべき範囲は広く、国内外の動向もかなり日々激しく変化している(令和2年現在)
(*) 令和3年の改正も予定されている

原則

現在、多くの領域で共通見解としてみなされているアプローチとして、Ann Cavoukian博士が提唱した"**Privacy by Design**" (PbD)が挙げられる。
"PbDとは、「プライバシー侵害のリスクを低減するために、システムの開発においてプロアクティブ(proactive 事前)にプライバシー対策を考慮し、企画から保守段階までのシステムライフサイクルで一貫した取り組みを行うこと」である。" (Wikipediaより)
ここでは、その7つの原則を例示する。



また、個人に関するデータを取得するにあたっては、コレクションミニマイゼーション(収集の最小化)、データミニマイゼーション(データの最小化)を**必ず**行わなくてはならない。対象が人であるサービスにおいて、ほぼ全ての情報は個人に関係する情報となる。その為、個人情報保護法の法的定義に縛られずに、慎重にデータの取り扱いを設計するべきである。

参考資料

- 「DX時代における企業のプライバシーガバナンスガイドブックver1.0」
- (消費者向けオンラインサービスにおける通知と同意・選択に関する国際標準化等の状況に係る調査事業)報告書
- Privacy By Design7つの基本原則
- https://www.tpdms.jp/file/20181227-1presentation.pdf (ミニマイゼーションのいい資料がなかった…)