ロボアプリ標準ガイドライン

改訂履歴

バージョン	日付	変更内容
1.1.0	2018/08/30	「3.1.4.1. 開発」内の対象NAOqiバージョンに関する表記を変更
1.0.0	2017/03/29	新規作成

本ガイドラインに登場するPepperの構成ソフトウェアは、出版・更新時点における最新リリースバージョンを元に記載している。

はじめに

本ガイドラインはPepper for Biz（以下、Pepper）におけるロボアプリ開発の理解促進と技術向上を目的とし、開発会社に対して標準化したナレッジを提供するものである。ロボアプリ開発のライフサイクルプロセスを、企画・要件定義、設計、開発・テスト、リリース、運用の5フェーズ、25プロセスに細分化し、各プロセスで実施すべき作業内容を方法論としてまとめあげた。

本ガイドラインは開発会社のみならずお客様自身にも理解して頂きたい内容になっている。各開発会社とお客様の双方が本ガイドラインを通して、Pepperの開発案件が円滑かつ効率的に推進する共通のコミュニケーションツールとして活用することで、品質の高いロボアプリが開発されることが期待される。

本ガイドラインの特徴

- 開発者として「何を」すべきかを記載し、「どのように」実施するかは基本的に触れていない。なぜなら「何を」は品質改善として推奨されるべき事柄であるが、「どのように」は開発会社及びプロジェクトの特性を考慮すべきだからである。従って、管理ツール・管理手法・作成資料（成果物）の名称・形式・書式は本書では問わないようにしている。
- 「どの程度（深さ）」に関しては、プロジェクト難易度や社内規定、開発会社の特性を考慮し基本的に触れていない。

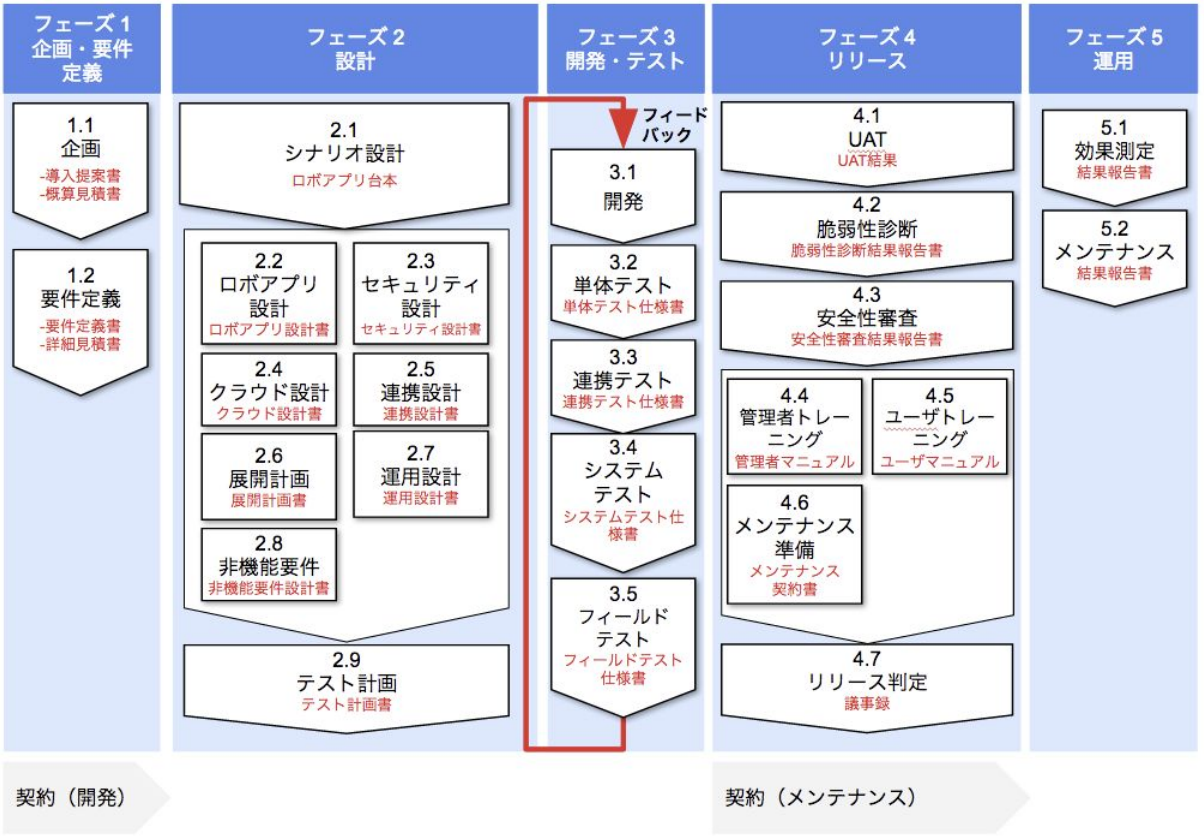
本ガイドライン活用によるゴール

開発会社自身が、以下の項目を実現できることをゴールとしている。

- お客様の課題やニーズを把握し、Pepperで実現できる最適な企画を提案できる。そのためにPepperの最新機能を理解し、運用後の効果測定まで意識してお客様の期待に沿う説明ができる。（企画・要件定義）
- 企画・要件定義から業務フロー、シナリオへの落とし込みができる。またシナリオからロボアプリ設計書の作成はもちろん、Pepperの仕様・制限事項・注意事項を十分理解し、セキュリティ、クラウドや連携まで一貫して設計できる。（設計）
- Pepperの開発環境やガイドラインを理解し、品質の高いロボアプリの開発ができる。また設計品質を向上させるテスト項目の作成、開発品質を向上させるテスト実施ができる。（開発・テスト）
- Pepper運用管理者・現場オペレータへのトレーニングおよび運用設計等を含め、必要な稼動準備を設計・実施できる。（リリース）
- 導入提案書、メンテナンス契約に基づいた運用や効果測定ができる。（運用）

プロセスマップ

ロボアプリ標準ガイドラインは、大きく5つのフェーズに分けられる。1. 企画・要件定義、2. 設計、3. 開発・テスト、4. リリース、5. 運用である。全体を図示すると以下のようになる。



矢印はプロセスの流れと依存関係を示している。

1. 企画・要件定義

企画・要件定義は、お客様とのコミュニケーションを基に提案し、実装されるべきシステムを形成するフェーズである。

企画段階では顧客価値を最大化するため、お客様の業務課題を掘り起こし、戦略的企画として考案する。そして導入目標（KGI・KPI）を設定し合意する。

本フェーズは「1.1 企画」、「1.2 要件定義」の2プロセスとなっている。

1.1. 企画

1.1.1. プロセス概要

企画プロセスでは、主に以下の作業を想定している。

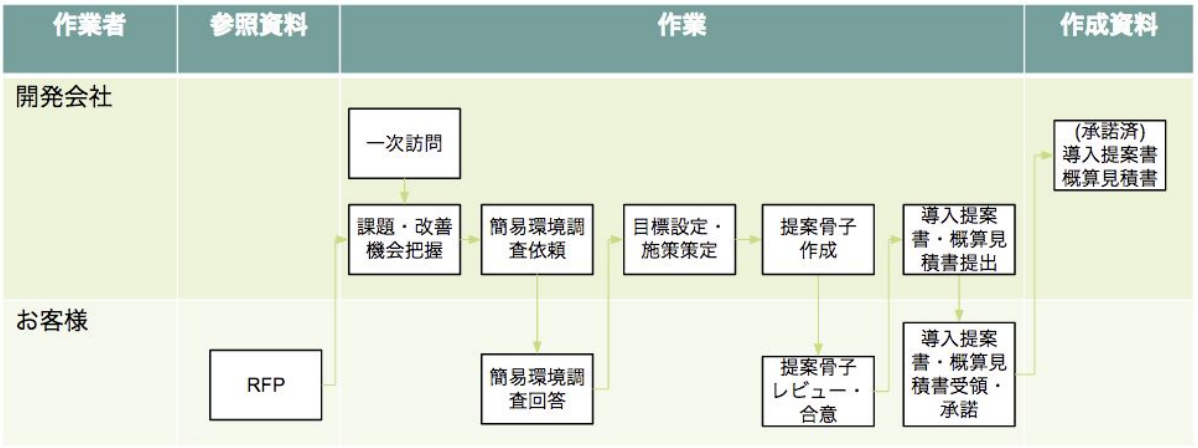
〈主な作業〉

- 施策策定前に、現地訪問前の環境確認として、お客様へ質問形式（Q&A）による簡易環境調査を実施する。チェックリストをお客様に提示し、導入にあたって障壁がないことを確認する。導入の障壁となりそうな回答は、対応を検討する。
- お客様の事前リサーチ、一次訪問もしくは提案依頼書（以下RFP）などから、お客様が抱えるビジネス課題やニーズを把握し、プロジェクトの導入効果を高める施策を策定する。その後提案骨子を作成し、お客様と合意を得る。
- 提案骨子を基に概算スケジュール、概算見積書を作成した上で導入提案書を提出する。

1.1.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	RFP
3	作成資料	導入提案書、概算見積書
4	完了基準	お客様による導入提案書、概算見積書の承諾
5	後続プロセス	要件定義

1.1.3. 作業の依存関係（企画）



1.1.4. 作業詳細

1.1.4.1. 提案依頼書（以下RFP）

お客様からRFPを受領した場合、RFPを基に提案を行う。

1.1.4.2. 一次訪問

お客様の課題やニーズを掘り起こし、Pepper導入によって実現すべき最終的なゴールを提案するための材料を収集する。

訪問にあたっては、業務課題やニーズの掘り起こし、新企画提案の可否に繋がるため、訪問会社のWEBサイトやプレスリリース等公開情報から中長期的な目標、指針を把握し、お客様の目指すものの概要を把握しておく。

お客様にPepperでしか実現できないことを理解してもらうために、事例や動画、絵コンテ、モック等を用いて説明する。お客様に具体的な導入後のイメージが伝わり、どのような業務に利用し、どのような制限・制約があるのか想定してもらいやすくなり、開発の手戻りを防ぐことに繋がる。

また、Pepperのバージョンごとの機能差分、最新機能や新企画を訪問前に理解しておくことでPepperの説明と今後の提案に支障が生じないようにする。

提案に繋がる課題やニーズの掘り起こしだけでなく、運用開始時期、予算感、決裁者の確認や導入後の運用に関する説明及び確認を行う。

以下、Pepperで重要な事項を確認事項としてまとめた。

確認事項（企画）		確認
1	Pepperで実現できること、できないことの基本的な内容説明を行い理解を得ているか（Pepperに対する行き過ぎた期待を適正化する）	
2	業務課題ヒアリングを実施できているか	
3	新企画がある場合、説明を行い理解を得ているか	

4	お客様がどのような業務に利用するか想定できているか	
5	お客様の実現したいことを理解しているか	
6	お客様の予算感を把握しているか	
7	予算に運用フェーズを考慮しているか	
8	お客様側の決裁者を特定しているか	
9	導入後の効果測定の必要性を説明し、実施の理解を得ているか	
10	お客様側で想定される運用業務（Pepperの機体管理等）の説明と、必要性の理解を得たか	
11	開発会社側で想定される運用業務（ヘルプデスク、障害対応、バージョンアップ等）の説明と、必要性の理解を得たか	
12	お客様側の運用体制を確保できるか	
13	運用開始時期を明確にしているか	

1.1.4.3. 課題・改善機会把握

お客様の目指すものから目的を整理する。目的を基に想定される課題を設定する。

1.1.4.4. 簡易環境調査依頼

Pepperの導入を円滑に進めるため、お客様側でシステムが導入可能か確認をお願いする。確認項目は以下の通りである。

確認事項（企画）		確認
1	屋内でPepperを利用するか	
2	Pepper及びPepperの梱包箱の保管スペースを確保できているか	
3	Pepperの稼動スペースを確保できているか	
4	電源の確保を行っているか	
5	電源（電圧）が安定しているか	
6	機器間で電波干渉が起こらない環境か（例：Wi-Fiは混線しやすい）	
7	インターネットと接続されたWi-Fiネットワーク環境が構築可能か	
8	DHCPによるIPアドレス取得が可能か（Pepperは原則固定IPアドレスを設定できない）	
9	什器、ポスター、ノボリ設置を予定しているか	
10	子どもの利用が少ない環境か	
11	ペット等、動物が少ない環境か	

12	大きな騒音・ノイズが頻発しない環境か	
13	光量の多い光源が設置されていない環境か	
14	Pepper が揺れない環境か	
15	煙・臭いの充満しない環境か	
16	水に濡れにくい・多湿ではない環境か	
17	極度の高温・低温（5～35度を外れる）ではない環境か	
18	火気のない環境か	
19	通行の障害とならない設置環境か	
20	高い清潔性が求められない環境か	
21	想定される連続稼動時間はどの程度か ①3時間／日、②6時間／日、③週1回程度／日、またはそれ以下、④その他など	
22	充電器に繋いだままの運用を予定しているか	
23	床面に斜度・反射・段差はないか	
24	デコレーションを予定しているか（ロボデコレーション）	

1.1.4.5. 簡易環境調査回答

お客様側で実施した簡易環境調査からPepperの導入にあたり障壁がないことを確認する。導入の障壁となりそうな回答は、対応を検討する。今後開発するロボアプリが、実際の環境で利用できないことを防ぐ重要なタスクとなる。

〈導入の障壁となりそうな回答結果と、対応の方向性〉

回答結果		具体例	発生する不具合（例）	対応の方向性（例）
1	屋外環境	雨風、砂塵、海水（波しぶき・塩害）海岸付近、船上、雪・氷	本体の破損、異物混入によるハードウェア不具合 禁止チャネル使用（5GHz帯のWi-Fiは屋外の場合、使用禁止となるチャネルが存在）	・出来る限り壁ぎわ、屋根のある場所に設置する。 ・禁止チャネル以外のチャネルを利用する。
2	電源（電圧）が安定しない	不安定な電源（車両のエンジン等、不安定なインバーターを利用した電源）	ディスプレイの動作不良、誤動作等	・運用中は充電しないでバッテリーを利用。運用時間外で充電する。（電源が安定した環境を確保する）

3	機器間の電波干渉	船舶無線・医療機器への電波干渉、ラジオの受信障害	船舶無線の誤作動、ラジオ等の受信障害	・電波影響を受ける場合：心配な機器があれば事前に検証する ・Pepperの電波が影響する場合：VCCI取得の為、問題ない
4	ネットワーク接続が十分でない	Wi-Fi接続が不十分な環境（ルーターとの距離が遠い、受信時の障害物）船上、離島、山間地域、オフライン環境	アプリ・ソフトウェアの動作不良	・ Wi-Fi接続の十分な環境で事前にアプリの起動・利用を行うことで必要なリソースをダウンロードさせておく。
5	子どもが多い	ショッピングモール等	指の故障	・人が入れないように什器を設置する。
6	動物が多い	ペットショップ、動物病院の待合室等	異物混入によるハードウェア不具合 臭いの付着	・動物が入れないように什器を設置する。 ・臭いがこもらないように換気する。 ・空調を効かせる。
7	騒音・ノイズが頻発する	他のお客様の話し声、館内放送、音楽演奏	音声認識が機能しない、誤認識	・ディスプレイで音声認識と同様の対応ができるようにし、音声認識をオフにする。
8	光量の多い光源が設置されている	スポットライトの設置、外光（直射日光）	センサー誤作動、本体の日焼け	・光や外光を受けないよう光避け、日差し避けを用意する。
9	Pepperが揺れる	輸送機器内での利用（客船・バス・電車上・飛行機等）	センサー誤作動、転倒による破損	・ （転倒防止のために）固定させる土台を使用する。
10	煙・臭いが充満する	喫煙スペース、飲食店厨房、フロアで煙・臭いが強い環境	黄ばみ、臭いの付着	・臭いがこもらないように換気する。 ・予備のPepperを用意する。 ・煙の発生源から離して設置する。
11	水・水滴・多湿・結露が発生する	公園の噴水付近、イベント演出でのミスト、サウナ等	サビによる腐食、可動部の不具合（異音）	・結露・多湿は避ける。（除湿を行なう） ・結露の発生する環境では使わない。（回避策無し）
12	高温・低温となる（5～35度を外れる環境）	温度管理のない倉庫等	可動部の不具合	・空調を効かせる。

			（低温時：稼動部グリスが硬くなる）	
13	火気がある	飲食店等の調理場、製造工場内	Pepperの高温化による不具合	・火気から離して設置する。（基本的に使用不可） ・目のセンサーが火を検知しないように設置する。
14	通行の障害となる設置環境	人の通行を妨げる狭い通路上等	本体破損、ハードウェアエラー	・できるだけ人の導線に被らないように置く。
15	高い清潔性が求められる	病院、研究室、製造ライン等	機体の汚れ	・指定のクリーナーで清掃する。 ・目的別（殺菌・清掃）にクリーナーを利用する。

1.1.4.6. 目標設定・施策策定

目標設定及び施策を策定する。

KGIとは、プロジェクトの最終ゴールとなる達成すべき目標のことである。一方、KPIは、KGIを達成するための過程で計測する中間目標のことである。

不必要な開発の抑制のために、以下の導入難易度に応じてできることを整理し、段階的に進めることでKGIを達成できるか検討する。またその際には、各段階でKPIを設定し、効果測定ができるようにする。

〈導入難易度の段階例（開発を伴うPepper導入案件）〉

第1段階：お仕事かんたん生成（BizPack）及び無料ロボアプリで実現できること
第2段階：ロボアプリマーケットの有料ロボアプリで実現できること
第3段階：ロボアプリ個別開発を伴うもの
第4段階：外部システム・ハードウェアとの連携を伴うもの

〈段階的な導入例〉

KGI：顧客生涯価値20%向上（前年比）

施策		概要	導入難易度	KPI
1	ショッピングアドバイザー	Pepperが売り場で呼び込み・接客をしながら、お客様の要望・嗜好をヒアリングし、おすすめの商品をご案内する。	第1段階（BizPack）	集客数10%向上（前期比） 購入単価5%円向上（前期比）
2	ヒアリング情報連携	Pepperがお客様からヒアリングした情報を店舗スタッフの手元の端末に連携し、店舗スタッフが接客に利用する。	第3段階（ロボアプリ個別開発）	販売員の接客業務効率化で接客数10%向上（前期比） クロスセルによる購入単価5%向上（前期比）
3	Pepperレジ	Pepperが在庫照会を行い、欠品時にECでの購入ができるようにして、顧客満足度の向上と	第4段階（外部システム連携）	販売機会損失ゼロ

		機会損失を低減する。		

目標設定、施策策定は、人型ロボットがもたらす商品価値を考慮し策定する。（「なぜPepperでなければいけないのか」「Pepperだからこそ実現できること」を策定する。）

〈人型ロボットがもたらす価値例〉

カテゴリ		例
1	単純作業	呼び込み、クーポン、夜間警備、待ち札、フロアマップ、レク、徘徊見守り
2	心理ハードル低下（人が伝えにくい事柄）	中古車査定、ヒアリング、レコメンド、簡単検診、個人情報を収集する
3	機能拡張性（クラウド・IoT・AI連携）	外国語、ポイントカード、FAQ、翻訳、多言語PR、顔認証、プリンタ連携、電話・SMS連携、パスポートスキャナ連携、ディスプレイ連携、テレプレゼンス
4	情報収集・分析	BI、ビッグデータ連携

目標設定は以下の視点で適切な設定ができているか評価し、後続プロセスである効果測定を実施できるようにする。

〈KGI・KPI設定の評価軸〉
- 具体的であるか
- 計測可能であるか
- 達成に向けて実現可能性が高いか
- 期限が設定可能か

お客様に以下の確認を行い、上記評価軸となるKGI・KPI設定を行う。

〈確認項目〉
- お客様が持っているデータで、評価軸として出せる数値・データは何か
- どの数値、データが評価されるか

上記を確認することで評価対象が具体的となり、過去数値との比較ができるため計測及び期限の設定が可能となる。具合的な設定例は以下となる。

〈設定例〉
- 売上10%アップ（前年比）、来店数10%アップ（前年比）、発券数10%アップ（前年比）
    - コスト削減等、数値の減少による評価は容易ではないものの、来店数や発券数等、数値の上昇による評価は設定しやすい。インタラクション分析を利用する場合は、信頼性向上のためダブルチェックとして現場で直接計測するか否かを検討する。
- 受付未対応ゼロ（前年比）、商談機会10%アップ（前月比）

- 例として平均待ち時間5分削減といった設定は可能であるが、大幅な改善は容易ではなく、大抵、待ち時間の解消は体感時間での評価となる。従って、待ち札の発券をしている場合は、受付未対応ゼロ、商談機会10%アップ等、機会損失の改善観点で設定すると効果測定が行いやすくなる。

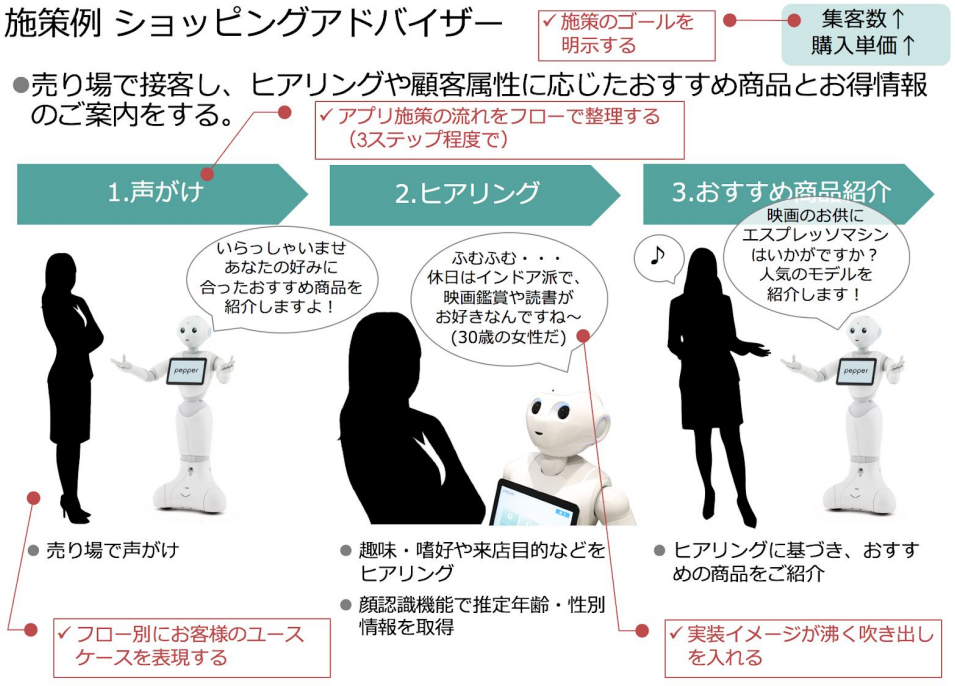
- 提案件数10%アップ（前月比）
  - 店舗スタッフが店頭で行なっていたオススメ商品の提案をPepperが行う等、業務代替、業務効率化を行なった場合、今まで作業を行なっていたスタッフが空いた時間で何を行なったのか、何を行うのかが定義されていないと業務効率化の評価に繋がらない。そのため、空いた時間で何を行うのか、事前に定義し、業務代替、業務効率化が行われるようにする。

確認事項（企画）		確認
1	お仕事かんたん生成（BizPack）及び無料ロボアプリ、有料ロボアプリ、個別アプリといった各開発の切り分けをできているか（切り分けた上で実現可否の検討、施策策定ができているか）	
2	評価測定可能なKGI・KPIを設定できているか	
3	導入難易度に応じたKPIを設定できているか	

1.1.4.7. 提案骨子作成

一次訪問でお客様からヒアリングした情報、あるいはRFPに沿った形で提案骨子を作成する。課題に対する対応策は、大まかなユースケース（Pepperとエンドユーザとのやりとり）を用意する。（ユースケースによりPepperが具体的に何を行うのか、導入イメージが沸きやすくなる。）

〈ユースケース例〉



また、課題に対する対応策をどのように実施・実現していくのか、時系列にまとめたロードマップを用意し、設定した目標の測定タイミングを提示する。作成したロードマップを基に、概算スケジュールを作成する。

〈ロードマップ作成によるメリット〉

- 複数施策での実施タイミング、効果測定タイミングを整理できる（同時並行による混乱、実現タイミングについてお客様との認識相違を回避できる）
- 「現在Pepperが実現できること」と「将来できるようになること」を分けて提示できる（対象範囲を明確化し、今後の継続的な取引及び提案が可能になる）

提案内容だけでなく、導入・運用に関わる以下の内容を整理する。

〈整理内容〉

- 導入・運用上での注意ポイント
    - 例：オンサイト初期設定、キッティング、個社窓口、その他オプション等の対応サービス、複数台Pepperを導入する場合の展開計画
- 利用にあたっての前提事項・制約事項
    - 例：屋内外、スペース、常時電源接続、常時Wi-Fi接続等
- 免責事項
    - 1. 契約に含まれる利用エリアから遠距離・海外等での利用を行う場合、お客様責任になること
    - 2. 納品後の通常利用でPepperが危害を与えてしまった場合、開発会社はその責を負わないこと
    - 3. 納品後お客様自身で修正・変更を行った場合、ロボアプリ安全性審査を実施できないこと
    - 4. 審査済みロボアプリが故障の起因となった場合、お客様が負担する修理費用を免除とする免責事項がなくなること

プロジェクト計画作成

お客様が希望する納期で、お客様が求める品質のロボアプリを効率よく開発するために、以下を整理したプロジェクト計画を作成する。

〈計画項目〉

項目		実施事項
1	システム構成図	外部連携のあるなしにかかわらず用意する。外部連携がない場合は、Pepperとインターネットに繋がっているWi-Fiアクセスポイントを記載する。
2	マスタスケジュール	1. プロジェクトのマイルストーンを設定し、タスクを時系列で整理する。 2. 各プロセス、マイルストーンでチェックすべき項目を設定する。 3. 稼動後の保守作業や効果測定も含めたスケジュールを引く。 4. 定例会議の予定や日別のタスクスケジュールを線表形式で記載し、進捗を管理する。

3	タスクと役割分担	開発会社とお客様の役割を整理する。
4	タスクとアウトプット	タスクの達成基準を明記し、タスク完了時点の作成資料を記載する。
5	プロジェクト推進体制	プロジェクト関係者をお客様・開発会社に分けて記載する。 プロジェクトメンバのアサインでは各人の役割を定義する。
6	依頼事項	プロジェクト推進上、必要となる依頼事項を記載する。 （例：定例会議体の調整、メンバのアサイン等）

提案骨子の作成と共に、お客様への提案にあたって必要なメンバの確保もしくは確保の見通しを立てる為に、関係各所と調整を行う。

またマスタスケジュール作成にあたっては以下の点に注意する。

〈注意点〉

- 脆弱性診断、ロボアプリ安全性審査の期間を確保できているか
  審査期間の確保だけでなく、設計・開発・テストが不十分で審査が不合格となった場合、その対応（修正・各種再テスト実施・再審査）でリリーススケジュールに影響が生じる場合がある

確認事項（企画）		確認
1	課題、ニーズに対する対応案を出せているか	
2	「実現できること」「実現できないこと」を整理できており、「実現できないこと」に対して代替案を検討できているか	
3	業務代替の場合、お客様の業務のどの部分が変わるか、明確になっているか	
4	導入・展開のロードマップを提示できているか	
5	ロードマップに沿ったマスタスケジュールを記載しているか	
6	導入までのプロセスを具体的に定義できているか 例：開発期間、各種テストプロセス	
7	導入・運用上での注意ポイントを説明し理解を得ているか	
8	利用にあたっての前提事項（制約事項）を説明し理解を得ているか	
9	免責事項を説明し理解を得ているか	
10	お客様への提案にあたって必要なメンバの確保もしくは確保の見通しは立っているか	

1.1.4.8. 提案骨子レビュー・合意

作成した提案骨子をお客様に説明し、レビュー結果を得る。

1.1.4.9. 導入提案書・概算見積書提出

この段階では開発にかかる費用を決定するのは難しい。しかし、お客様の予算感を確認するため概算レベルで見積を作成することによって費用感を把握し、詳細見積の段階までに精度を高めることに繋がる。

概算見積には、展開アプローチを含むマスタスケジュール及び前提条件を含めることで、詳細化前に必要なタスクが洗い出され、見積の精度が向上とお客様との認識齟齬を防ぐことにつながる。

また納品物を明記することで追加のドキュメント作成が発生することを防ぐようにする。納品サンプルを提示し、お客様との認識の相違がないか確認することでトラブル撲滅にも繋がる。

〈見積作成時の前提条件例〉

- クラウド使用料等が必要となる場合は別途お見積とさせて頂きます。
- ロボアプリ配信及びお仕事かんたん生成の設定は、貴社での実施を想定しております。
- 開発に必要なPepper及び開発関連のツールにつきましては弊社にて調達致します。その他有料ツール類の調達が必要となる場合は、お客様にてご用意頂きます。
- PepperのソフトウェアバージョンはNAOqi OS 2.4.3の見積となっています。NAOqi OSのバージョンアップが発生した際の対応に必要な費用は、別途お見積とさせて頂きます。
- 保守運用に必要な費用は、別途お見積とさせて頂きます。

確認事項（企画）		確認
1	見積の前提条件を記載しているか	
2	納品物を記載しているか	
3	概算見積書の社内チェック・承認を完了しているか	

1.1.4.10. 導入提案書・概算見積書受領・承諾

お客様に導入提案書及び概算見積書を提出し承諾を得る。承諾済み導入提案書及び概算見積書をもって後続プロセス「要件定義」へと進む。

1.2. 要件定義

1.2.1. プロセス概要

要件定義プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 導入提案書を基に、ロボアプリ要件一覧を作成し、for Bizテンプレートを選定する。

- 導入提案書、ロボアプリ要件一覧、For Bizテンプレートを基に、ロボアプリ全体フローを作成する。
- ロボアプリ全体フローを確認し、要件一覧、for Bizテンプレートと共に要件定義書を作成し、お客様に提出する。
- お客様から要件定義書の承認を得て、詳細見積書を作成しお客様に提出する。
- お客様から提出した詳細見積書の承諾を得る。

1.2.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	導入提案書
3	作成資料	要件定義書、詳細見積書
4	完了基準	お客様による要件定義書の承認、詳細見積書の承諾
5	後続プロセス	シナリオ設計、セキュリティ設計、クラウド設計、連携設計、展開計画、運用設計、非機能要件

1.2.3. 作業の依存関係（要件定義）

1.2.4. 作業詳細

1.2.4.1. ロボアプリ要件一覧作成

企画プロセスでお客様の合意を得た導入提案書を基に、要件一覧表を作成する。
ロボアプリ要件一覧では以下を記載する。

〈記載内容〉

- 要件と狙い：提案段階で挙げられている機能要件と、これを実現することで達成したいことを列挙する
- 実現手段（開発範囲の明確化）：BizPackで実現できるのか、個別開発が必要なのか明示する
- 実現案詳細：具体的な実現方法を記載する

機能要件を支えるパフォーマンス・セキュリティ・トレーニングや運用に関わる要件も整理する。個人情報・認証情報等のセンシティブなデータを扱う場合は、暗号化の方式・秘密鍵の格納方法等、セキュリティ観点での要件も整理する。

提案書で作成したシステム構成図を基に以下を確認する。
以下を確認することで、システム構成における制約・前提となる条件が整理されるため、そこから開発にあたって課題や問題がないか確認する。

確認事項（要件定義）		確認
1	対象システム・機器・サービスは全て示しているか 例：Pepper、Watson、iPad等の端末、ネットワーク、クラウド等	
2	連携する外部システム・外部機器の連携方式、動作環境（バージョン）を記載しているか 例：iPad（ iOSバージョン10以上）	
3	複数のシステム・機器・サービスが関連する場合、相互関連性、連携方式、処理内容の概要を記載しているか	

Pepper利用が初めて（Pepper for Biz新規契約）の場合、トレーニングでお仕事かんたん生成の使い方等、開発対象外のトレーニングを求められる可能性が高く、実施直前で問題とならないよう事前に要件として必要可否を確認する。

確認事項（要件定義）		確認
1	開発範囲を明確にできているか	
2	ロボアプリの開発要件を一覧表にまとめてあるか	
3	開発要件でセンシティブなデータを扱う場合、セキュリティの観点での考慮があるか	
4	トレーニングの対象範囲を明確にしたか	

1.2.4.2. for Bizテンプレート選定

BizPackで利用するfor Bizテンプレートを選択する。テンプレートは接客、受付、フリーの3つである。

〈テンプレート〉

名称		概要
1	接客	呼び込み、挨拶、つかみトーク、メニュー、締めトークの流れで構成されている。店頭でお客様の呼び込みや、お店の商品紹介を想定している。
2	受付	受付テンプレートは挨拶、受付、メニュー、締めトークの流れで構成されている。来社されたお客さまへ、担当者や担当部署の連絡先をご案内、会社のサービス紹介等を行うことを想定している。
3	フリー	呼び込み、挨拶、メニュー、締めトークの流れで構成されている。接客・受付

		テンプレートよりも自由度の高いカスタマイズが可能である。

〈各テンプレートの仕様〉

テンプレート		発話機能の仕様			メニューに追加可能なタスク
		呼び込み	挨拶	つかみ・受付	
1	接客	10種類のプリセットトーク × 3種類のフリートーク（最大50文字まで）	法人名・店舗名（最大30文字まで） × 6種類のプリセットトーク	複数種類のプリセットトーク（天気や季節、日常会話等）	ご案内 商品紹介 性格診断商品紹介 アンケート 遊ぶ マイアプリ
2	受付		法人名・店舗名（最大30文字まで） × 3種類のプリセットトーク	受付業務向けのプリセットトーク	ご案内 サービス紹介 連絡先検索 遊ぶ マイアプリ
3	フリー	10種類のフリートーク（最大50文字まで）	10種類のフリートーク（最大50文字まで）		プレゼン（=商品紹介） 2択診断 アンケート 遊ぶ マイアプリ

1.2.4.3. ロボアプリ全体フロー確認

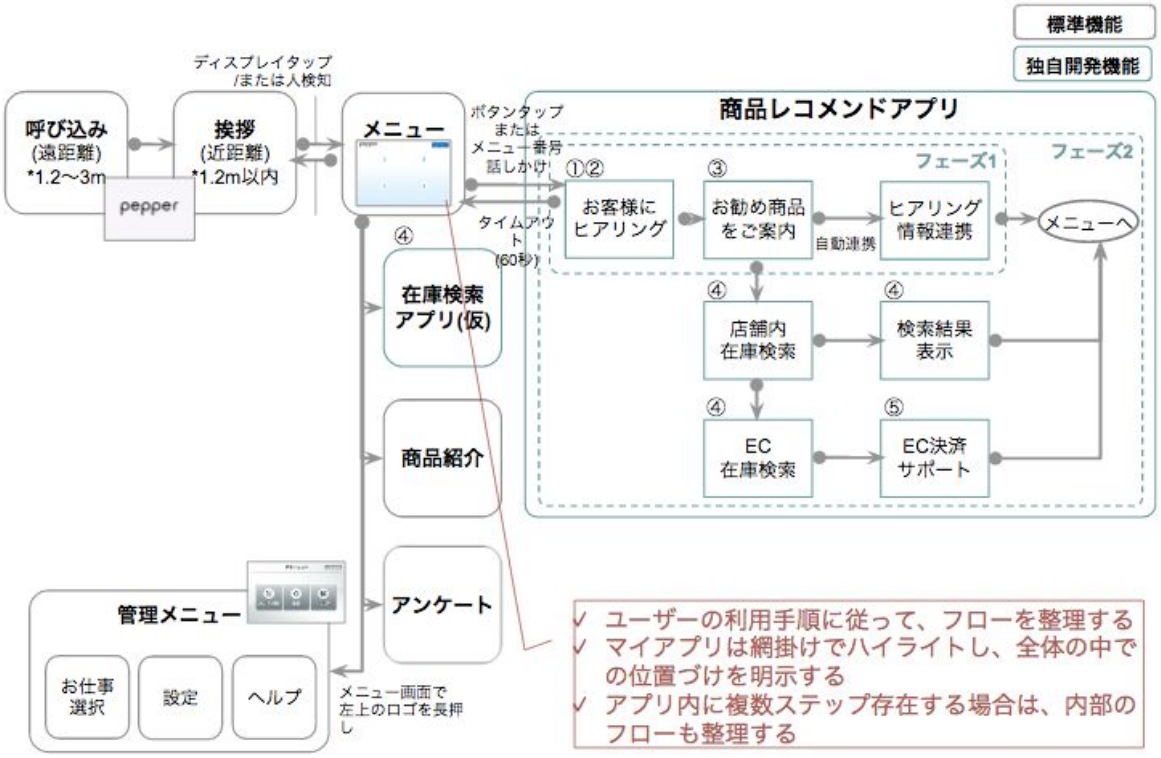
ロボアプリ全体のフロー図を作成し、お客様の確認を得る。作成目的は、実際の利用シーンを想定し、ロボアプリの進行に違和感がないか確認することである。

ロボアプリ全体フローは、開発するロボアプリの位置付けによって以下にある2種類（ロボアプリの進行に合わせたもの、業務フローベースで書き出されたもの）のいずれかで作成する。後続プロセスである設計フェーズで参照資料となるよう、フロー図は機能単位、もしくは作業単位で作成する。また、いずれの場合でも独自開発機能がある場合は該当する箇所を明示する。

〈ロボアプリ全体フローの種類〉

タイプ		特徴
1	ロボアプリ進行ベース	Pepperがシステムの主要構成要素となるもの、Pepperで完結しているシステムを使用する場合に最適。機能単位で作成し、ロボアプリ内で複数のステップが存在する場合は、その内容も明記する。
2	業務フローベース	外部システムとの連携が想定され、かつ、Pepperがシステムの主要構成要素とならないロボアプリを開発する場合に最適。複数システム・製品やサービスが関連する場合、相互関連性やエラー処理を含めた連携イメージを明記する。

〈ロボアプリ全体フロー例〉



画面要件の確認

後続プロセスのシナリオ設計ではなく、要件定義段階で画面要件を確認することで詳細見積（開発工数）の精度を向上させる。主な確認項目は以下となる。

〈確認項目〉

- 画面のイメージ（メッセージ文やボタン配置含む）
- 設計上のポイント（画面の処理内容等）
- 画面の遷移先
- 実装にあたっての検討・懸念事項
- レイアウト定義（ヘッダ（共通）、フッタ（共通）、ロボアプリ個別表示エリアでの表示内容等）
- 画面項目定義、単項目チェック内容（画面表示項目、I/O種別、コントロール（HTML部品）、文字表示（右・左・中央揃え）、表示タイミング、イベント一覧等。また、JavaScriptのみで入力チェックを実施する場合は、実施仕様（対象項目・チェック仕様を記載）

画面遷移や条件分岐が複雑な場合は、条件と画面遷移先を整理する。

1.2.4.4. 要件定義書提出

ロボアプリ要件一覧、BizPackテンプレート選定、ロボアプリ全体フローをまとめ、要件定義書としてお客様に提出する。

1.2.4.5. レビュー・承認

要件定義書をお客様に提出し、承認を得る。

1.2.4.6. 詳細見積書提出

概算見積書を踏まえ、精度を向上させた詳細見積書を提出する。お客様のスムーズな社内稟議（予算確保）、及び後続プロセスでの意図しない追加費用の発生を防ぐために、以下2点を明記する。

〈記載事項〉

- 見積の前提条件
    - 概算見積書記載の前提条件を確認し、修正・追加を行う
- 見積対象
    - フィールドテスト、メンテナンス準備の工数は運用設計後に判明するため、別途見積
    - ロボアプリ設計時に出てきた追加要件、追加機能は別途見積
    - 設計フェーズで見積前提条件・要件がお客様責任で変更された結果、再び企画・要件定義に戻った場合は、別途追加見積等

確認事項（要件定義）		確認
1	見積の前提条件を記載しているか	
2	見積対象を明確にしているか	
3	詳細見積書の社内チェック・承認を済ませているか	

1.2.4.7. 詳細見積書レビュー・承諾

詳細見積書をお客様に提出し、承認を得る。

ロボアプリ安全性審査の事前申請

開発・テストフェーズ終了後の申請では、その時点の審査状況によっては審査待ちになる可能性があるため、Pepper パートナー会員事務局へロボアプリ安全性審査[1]の事前申請をする（申請予定日・審査完了希望日を伝える）。（ただし審査完了希望日までの審査完了を保証するものではない）

確認事項（要件定義）		確認
1	詳細見積書（前提条件、対象含む）についてお客様の承諾を得ているか	
2	ロボアプリ安全性審査の事前申請を行ったか	

1.2.4.8. 契約締結（開発）

開発契約の締結を行う。本契約にはメンテナンスについての契約が含まれていないことを説明する。

[1] http://www.softbank.jp/robot/developer/dev-support/program/partner/benefit/#03 参照

確認事項（要件定義）		確認
1	導入提案書記載内容を反映しているか	
2	メンテナンス契約は後続プロセス（メンテナンス準備）で行うことを説明済みか	
3	開発会社内で、法務チェックを受けているか	

2. 設計

設計は、企画・要件定義フェーズで定義されたロボアプリに求められる機能要件を各設計書に落とし込むフェーズである。

ロボアプリ開発では機能要件を基にシナリオ化し、セリフ・モーションといったロボアプリ特有の設計を行う必要がある。また、ロボアプリはPepper単体で動作することもできるが、業務システム等と連携する場合、外部システムとインテグレーションの必要があり、連携方法等を定める連携設計の重要性が高まる。

設計フェーズの具体的な作業として企画・要件定義フェーズで定義された要件に基づき、ロボアプリ機能の段階的リリース、段階的な複数台のPepperのリリース等、展開計画を立案する。また、機能要件を支えるインフラに関連した非機能要件や、サポート体制等の運用設計を行い、各設計のテスト計画までを立案しておく。

本フェーズは「2.1 シナリオ設計」「2.2 ロボアプリ設計」「2.3 セキュリティ設計」「2.4 クラウド設計」「2.5 連携設計」「2.6 展開計画」「2.7 運用設計」「2.8 非機能要件」「2.9 テスト計画」の9プロセスとなっている。

2.1. シナリオ設計

2.1.1. プロセス概要

シナリオ設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

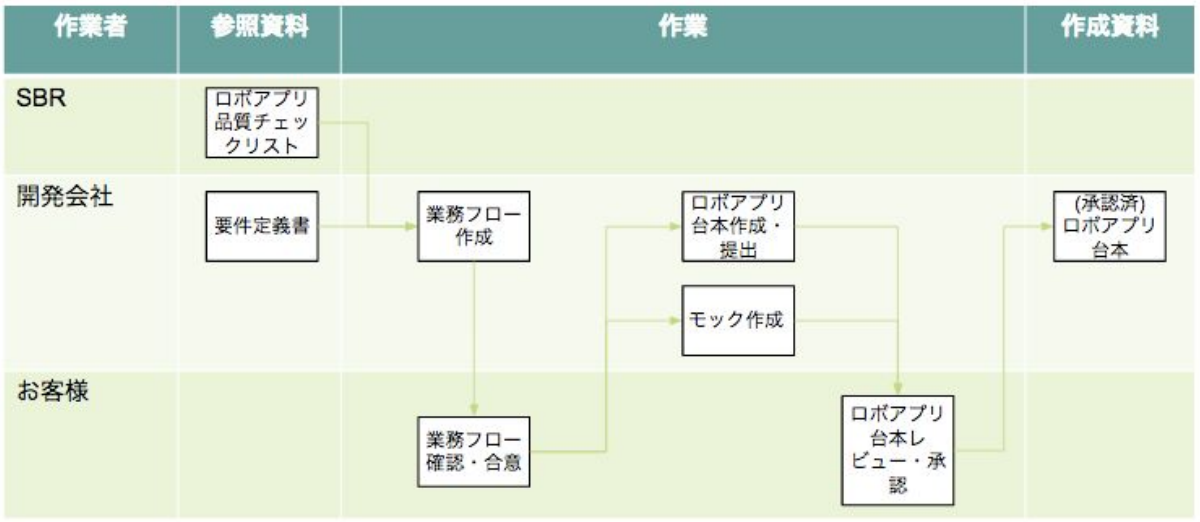
- 要件定義書を基にPepperのディスプレイ・セリフ・モーション等のロボアプリ仕様を台本（ストーリーボード）として作成し、お客様と合意する。
- 既存業務をPepperに代替する場合、どの作業が対象となるのかを明確にし前後の作業がわかる形で業務フローを記載する。
- ロボアプリ台本に書かれている流れをお客様と共有するため、モック（ロボアプリ）を作成する。

2.1.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	ロボアプリ品質チェックリスト[2]、要件定義書
3	作成資料	ロボアプリ台本
4	完了基準	お客様によるロボアプリ台本の承認
5	後続プロセス	ロボアプリ設計

[2] 旧称『マイアプリ開発ガイドライン』

2.1.2. 作業の依存関係（シナリオ設計）



2.1.3. 作業詳細

2.1.3.1. 業務フロー作成

要件定義書の内容に従い、お客様と開発会社双方で理解可能な業務フロー図を作成する。以下のように登場人物（アクター）、データ、ドキュメントを規定し作成する。

〈規定例〉

項目		内容
1	エンドユーザ	Pepperを操作する利用者（お客様のお客様）
2	Pepper	Pepper本体（ディスプレイ及びロボット）
3	現場オペレータ	Pepperの運用を行う現場スタッフ
4	スタッフ	Pepperと関わりを持つ既存業務従事者
5	Pepper運用管理者	Pepperの管理を行う現場管理者
6	業務システム	Pepperと連携するシステム
7	データ	Pepperと連携するシステムがやりとりするデータ 例：Pepperと連携しSMS出力するデータ等 Pepper内部のDB（SQLite）のスキーマまたはファイルに格納するデータ、JSONの構造
8	帳票、ドキュメント	連携するシステムが出力する帳票、ドキュメント等
9	クーポン	Pepperと連携しプリンタ等で出力するクーポン等の出力データ

業務フロー作成では、既存業務に精通しているお客様担当者のアサインがない場合、作成した業務フローに以下の問題が発生することが多い。

〈問題〉
- 対象業務の抜け漏れがある
- 業務フローが間違っている（作業の抜け漏れ、作業の繋がりが間違っている等）
- 作業の詳細がわからない（どのようなデータのやりとりがあるのか、担当者が誰か）

従って、既存業務に精通しているお客様担当者をアサインする。お客様担当者をアサインすることで得られるメリットは以下となる。

〈メリット〉
- 対象業務の抜け漏れを防ぐことができる
- 実現可能な業務フローが作成できる
  （現場環境や既存業務の制約、前提条件を考慮したフローにすることができる）
- 実施にあたり障壁となりそうな箇所、現場が懸念していることがわかり、事前に解決策、対応策を打つことができる

システム連携が含まれる場合や他部門・他部署が関与する等、関係者・連携システムが増えるに従い、現場の確認や現場による業務フロー検討が発生する。連携システムにかかわる業務から作成に着手し、登場人物（アクターの明確化）・データ・ドキュメントを規定（業務フロー図のフォーマットを確定）し作成する。
後続作業のロボアプリ台本と関連づけるために、作成時に以下の確認を行う。

〈確認項目〉
- Pepperの起動タイミングを確認
  Pepper for Bizでは起動トリガー条件や、起動センテンスが利用できなく、通常メニューからのみ起動が可能なことをお客様に説明し理解してもらう。

確認事項（シナリオ設計）		確認
1	業務フローはお客様が理解できる形式で記載されているか	
2	業務フローはソリューション中の登場人物・設備・システム・データが記載されているか	
3	業務フロー内でロボアプリの起動タイミング、起動方法は記載されているか	
4	業務フロー内に実現不可能な問題点・矛盾がないか	
5	Pepperと各担当の業務が明確に分離しているか	

2.1.3.2. 業務フロー確認・合意

作成した業務フローをお客様に確認し合意を得る。

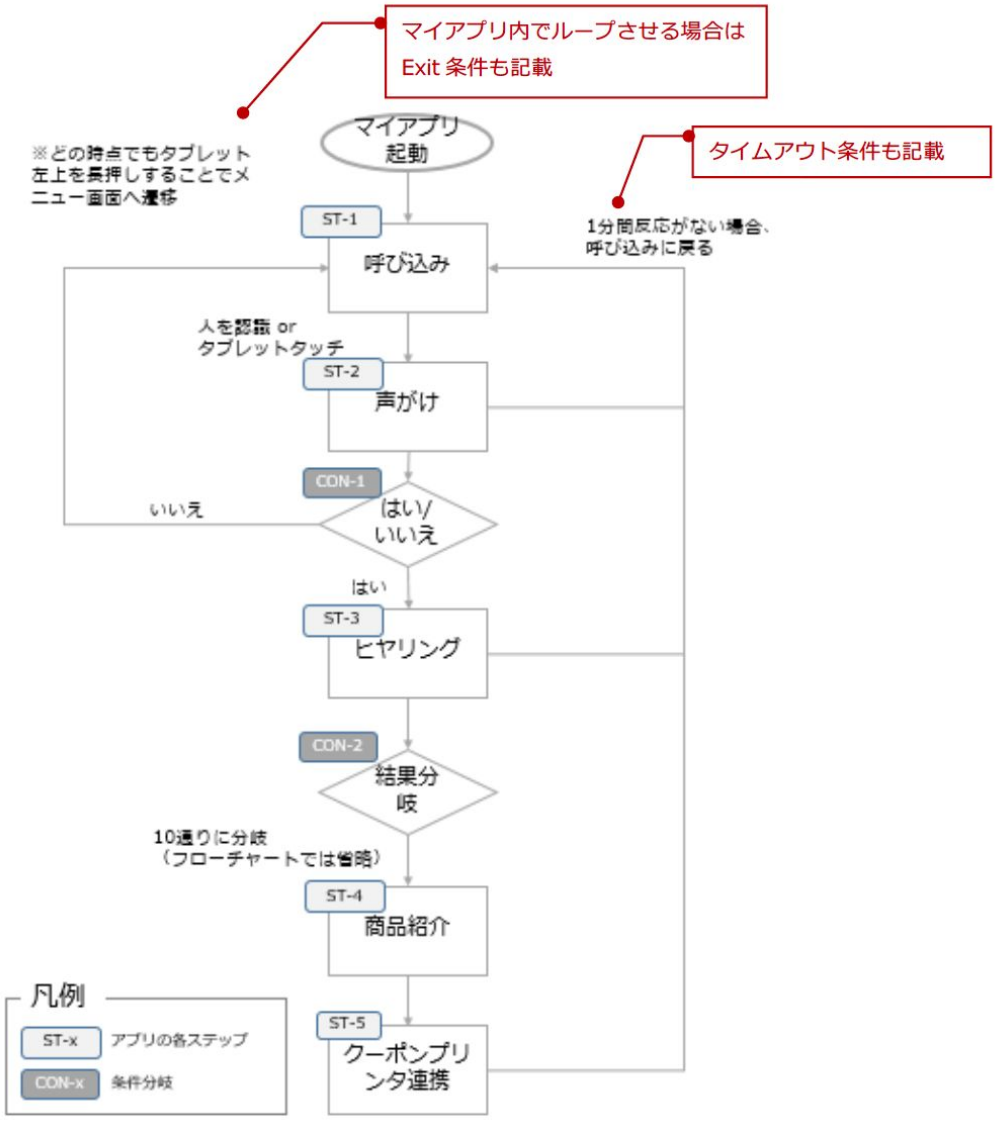
確認事項（シナリオ設計）	確認

1	業務に精通しているお客様から作成した業務フローの合意を得たか	

2.1.3.3. ロボアプリ台本作成・提出

ロボアプリ台本は、ロボアプリの進行を把握するための資料である。したがってPepperのディスプレイ・セリフ・モーション・連携システムとの連携場面等が並行記載される。台本の冒頭にフローチャートを用意し、全体の流れを把握する。ロボアプリ台本があることで、要件定義で挙げられたロボアプリ要件一覧にストーリー性が備わり、漠然としていたディスプレイ・セリフ・モーションの関連性がお客様に理解しやすい形に具体化される。

〈ロボアプリ台本例（フローチャート）〉



〈ロボアプリ台本例（台本）〉

ST-1：呼び込み

ディスプレイ	セリフ	モーション	ユーザ	外部連携

 ※各ステップでのディスプレイ画像とセリフを記載する。 ディスプレイ画像が準備できていない場合は、イメージがわくような絵や文言で代替する。	以下のセリフをランダムで発声 ・「いらっしゃいませ～。ボクがあなたにぴったりの商品を紹介しますよ！」 ・「こんにちは、Pepperです！ただいまクーポン配布中でーす」 ・「ボクがあなたの商品コンシェルジュになりますよー！」 ・「こんにちは！この店舗で働いている、Pepperです！」 ・「いくつかの質問に答えていただければ、あなたにおすすめの商品を紹介しますよ」	・両手を広げる	音声認識もしくはディスプレイタッチ	なし

シーンに合わせてPepperにセリフやポーズやモーションを取らせたい場合、セリフやポーズやモーションとディスプレイをシナリオに結びつけてロボアプリ台本に記載する。

作成時には、お仕事かんたん生成の動作フローを意識し、ロボアプリ品質チェックリストを準拠する。

加えて、過去プロジェクトを通して得た経験、知見を別途取りまとめ、ロボアプリ台本作成に活用すると、設計見直しの防止・バグ発生件数の削減に繋がる。

〈例〉

- 周囲の雑音が多い場所では、音声認識が期待通りに動作しないケースがある
  （そのため、音声認識を使用しないこと、もしくは音声認識のみでなく、ディスプレイ上での選択も補助的につけることを検討する）

確認事項（シナリオ設計）		確認
1	ロボアプリ台本は、エンドユーザ・Pepperディスプレイ・Pepperセリフ・Pepperモーションが同時に把握できるように構成されているか	
2	お仕事かんたん生成の動作フローを阻害しないか	
3	ロボアプリ品質チェックリストに準拠しているか	
4	ユーザ視点で問題はないか （例：シナリオが長すぎないか、内容が理解可能か、Pepperの質問は答えやすいか等）	

2.1.3.4. モック作成

設計フェーズで早期にロボアプリの完成イメージをお客様と共有するため、モックを作成する。モックとは、実際のロボアプリではなく、イメージ共有のための簡易的なロボアプリのことをいう。

Pepperとエンドユーザのやりとりは実際にロボアプリを見ないと要件を詰めることが難しい。例えば、呼び込み機能ではどう呼び込んだ方がいいのか、どのようなしゃべり方が伝わるのかは、実際に出来上がったロボアプリを見てからしか判断できない。

そのため、チューニング（要件定義・設計・開発・テスト・フィードバック）のサイクルを繰り返すことになるが、回数の増加は、開発・テストスケジュールの長期化による工数（コスト）増加を招く。

これを防ぐために、設計フェーズでモックを作成し、早い段階でお客様と共有することでチューニング期間の長期化を防ぐ。

2.1.3.5. ロボアプリ台本レビュー・承認

作成したロボアプリ台本をお客様に提出し、承認を得る。

2.2. ロボアプリ設計

2.2.1. プロセス概要

ロボアプリ設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 要件定義書、ロボアプリ台本を基に個別開発ロボアプリのロボアプリ設計書（基本設計書及び詳細設計書）を作成する。
- 設計内容がロボアプリ品質チェックリストに準拠していることを確認する。
- 設計書をお客様へ提示し、承認を得る。

2.2.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	ロボアプリ品質チェックリスト、要件定義書、ロボアプリ台本
3	作成資料	ロボアプリ設計書
4	完了基準	お客様によるロボアプリ設計書の承認
5	後続プロセス	テスト計画

2.2.3. 作業の依存関係（ロボアプリ設計）

2.2.4. 作業詳細

2.2.4.1. ロボアプリ設計書作成

要件定義書・ロボアプリ台本を基にロボアプリ設計書を作成する。ロボアプリ設計書は基本設計書及び詳細設計書から成り立ち、それぞれの差異は以下となる。

〈設計書の差異〉

種類	定義
基本設計書	お客様の見える箇所を設計する 要件定義書・ロボアプリ台本の記載内容「何を（開発対象）」「どこまで（処理内容・結果、性能）」「どのように（システム構成）」実現するか設計する。
詳細設計書	お客様の見えない箇所を設計する 基本設計書の記載内容を、開発者が実装できるようにコードもしくはChoregrapheダイアグラムに落とし込む。

基本設計書

基本設計書ではシステム構成や、開発すべきターゲット（NAOqi OSのバージョン、連携システムで使用するソフトウェア等）を明示する。前プロセス「シナリオ設計」で作成したロボアプリ台本によりPepperのディスプレイ・セリフ・モーションに表示されるUI等に関するフロー情報を盛り込む。

確認事項（ロボアプリ設計：基本設計）		確認
1	ロボアプリ品質チェックリストに準拠しているか	
2	システム構成図を作成しているか	
3	ロボアプリ情報を明記しているか 例：NAOqi OSバージョン、Pepperモデル等	

4	ディスプレイの画面レイアウトを記載しているか	
5	ディスプレイのUI遷移を明記しているか	
6	Pepperのセリフを記載しているか	
7	Pepperのモーションを記載しているか	
8	ロボアプリフローを記載しているか	
9	エラー処理時の分岐、例外処理、処理完了後の動作を含んでいるか	
10	後続プロセスとして実際に作成するビヘイビアを意識した記述となっているか	
11	ロボアプリを構成するファイル群（ライブラリやモジュール等）を定義できているか	

詳細設計書

詳細設計書ではロボアプリの動作単位であるビヘイビアからロボアプリで扱うデータ設計（ボックス間のデータ転送、不揮発性データの保存方法）、インタラクション分析でのデータ取得ポイントに至るまでを記述する。

ディスプレイ

ディスプレイに表示するUIパーツを確定する。ここで具体的な仕様を確定せず後続プロセスに委ねた場合、開発に余計な工数が発生するだけでなく、手戻りの可能性が極めて高くなるため注意する。

〈設計事項〉

- パーツID
- パーツサイズ
- パーツ座標
- デザイン
- 効果音

設計にあたりユニバーサルデザインの観点を活かし、誰に対しても使いやすいデザインを目指す。ディスプレイは画面構成だけでなく、応答速度にも配慮する。（例：HTML側でonClickイベントハンドラによる処理を行わず、jQueryライブラリを用いtouchイベントを捕捉する実装を明示する）。またタッチしたことが、感覚的に理解できるように効果音（クリック音等）を入れる。

連携・通信を伴う箇所

連携・通信が伴う箇所の設計は、「連携対象に接続する」と記述するにとどまらず、具体的な接続手順（プロトコル・プロセス）を明示する。具体的な記述がない場合、開発者ごとに異なる接続方法で実装を行ったり、意図している接続方法とは全く異ったその場限りの実装を行ったりする可能性がある。接続手順はシーケンス図と文章を用い明確化する。

ロジックの実装

処理ロジックは機体側あるいはディスプレイ側のどちらかに集約し、機体とディスプレイ間のメッセージングを減らすことで、処理の複雑化を防ぎ安定化につなげる。

ただし、ディスプレイ側はアプリのスタックリスクが存在するため、機体側との接続が失われた場合の対応処理を実装する。（例：ディスプレイから一定時間応答がない場合、ロボアプリを自動的に終了する）

ALMemoryの利用

ALMemoryは変数の格納、ロボアプリ間でのメッセージの受け渡しに使われるトランザクショナルメモリ機構である。詳細設計では、データの重複や意図しない変数間の衝突回避に繋がるため、利用キー名、及び役割を決定する。

ALMemoryの内容は他のロボアプリから任意アクセス（読み込み・書き込み）が可能なため、改ざんや漏洩などのリスクがある。セキュリティが求められるシーンでは、暗号化などの対応が必要となる。

インタラクション分析の利用

BizPackでは、インタラクション分析というロボアプリ利用に関する統計情報を取得する仕組みが用意されている。ロボアプリ側で特別な対応を行わなくても、起動回数等の基本情報は取得できるが、ロボアプリ側に［Get Mood］ボックスを組み込むことにより効果的な統計情報（例：クーポン発券数等）とそれを基にしたKPIまで取得できる。

なお、［Get Mood］ボックスを多数組み込むとインタラクション分析サーバに負荷をかけることに繋がるため、ロボアプリ品質チェックリストに示す上限を超えないようにする。

レビュー

設計者が多いと開発プロセスで重複開発に繋がるリスクがあるため、設計は少人数で行うか、少人数でない場合は全体を把握するレビュー者を置いて、重複・抜け漏れチェックを行う。

確認事項（ロボアプリ設計：詳細設計）		確認
1	ロボアプリ品質チェックリストに準拠しているか	
2	ビヘイビア・ボックス・クラス・メソッド・イベント・変数が役割に沿った名称になっているか（命名規則に準拠しているか）	
3	ビヘイビアの役割、依存関係が明確になっているか	
4	関連機能が一つのビヘイビア（もしくはサービスを含めたクラス）になっているか	
5	ビヘイビアに含まれる主なボックスの役割を明確にしているか	
6	ボックス間のデータの受け渡しを定義しているか	
7	重複するボックス・クラス・メソッドを作成していないか	
8	イベントの役割が明示化されているか	
9	画面項目に対するイベントを記載しているか	
10	データの形式もしくはデータ構造を記載しているか	
11	ALMemoryキーの使われ方を定義しているか 例：Pepperとディスプレイのメッセージング、特定変数の揮発データ共有用等	

12	不揮発性データ（ロボアプリ終了後ロボアプリ内外で利用するデータ）の保存方法を設計しているか	
13	コードの保守性を考慮した設計になっているか	
14	外部連携がある場合、その仕様（サービス・データ構造）を設計しているか	
15	ビヘイビア設計にインタラクション分析が含まれているか	
16	ディスプレイのコンテンツは、下記仕様に準拠しているか [Dimensions] 246 x 175 x 14.5mm [Media] Video codec H.264 / Video resolution 1280*800 / Bit rate 1Mbps以下 / Audio codec　AAC あるいは なし	
17	音声ファイルは、下記フォーマットとなっているか [Media] File format ogg	
18	開発者が開発できない記述、勝手に解釈してしまう箇所はないか 例：ボックス名が定義されておらず、開発者が自由に決められてしまう	
19	ロボアプリに含まれるボックス数は可能な限り少なくしているか（100未満推奨）	

2.2.4.2. ロボアプリ設計書レビュー・承認

ロボアプリ設計書をお客様に提出し、承認を得る。

2.3. セキュリティ設計

2.3.1. プロセス概要

セキュリティ設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 公表されているPepperセキュリティ仕様をお客様へ説明の上、外部システムとの連携方式やPepper上に保持する情報に関してセキュリティ設計を行う。
- 設計書をお客様へ提示し、承認を得る。

2.3.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	Pepperセキュリティ仕様、要件定義書
3	作成資料	セキュリティ設計書
4	完了基準	お客様によるセキュリティ設計書の承認
5	後続プロセス	テスト計画

2.3.3. 作業の依存関係（セキュリティ設計）

2.3.4. 作業詳細

2.3.4.1. Pepperセキュリティ仕様説明

ソフトバンクロボティクス（株）（以下SBR）が提供するPepperセキュリティ仕様の説明をお客様に行う。詳細は同シートに委ねるが、以下の点に特に留意が必要である。

〈留意点〉

- Pepperのみではセキュリティを担保できないため、インフラレベルで担保する
- セキュリティが求められる場面では情報階層に応じた暗号化を行う
- Pepperに対応するウィルス対策ソリューションは存在しない（外部システムにはウィルス対策ソリューションを導入可能）
- Pepperは自身のローカルIPアドレスを固定することはできない
    - IPアドレスを固定する要件がある場合、接続ネットワーク上でDHCPのサーバ側で設定を行う必要がある
- 認証情報、個人情報等セキュリティが求められるデータはPepperではなく、外部システムに保存する

2.3.4.2. セキュリティ設計書作成

Pepperセキュリティ仕様を基にセキュリティ設計書を作成する。

ネットワークセキュリティ

Pepperを企業で利用する際には高度なセキュリティ環境が必要になる。しかし、Pepperと同一ネットワークに接続された任意のコンピュータは、Pepperに認証情報なしでプログラムを転送・実行できるため、以下のネットワーク層及び物理層レベルでの対応を行い第三者の不正アクセスから守るよう設計する必要がある。

〈対応策〉

- Wi-Fiネットワークの暗号化
    - 高いセキュリティが期待できるWPA2-PSKを利用する（Web認証やEnterpriseに対応できない）。

    - セキュリティを高めたい場合、PSKに加えてMACアドレスによるアクセス制限を実施する（Pepper本体とディスプレイの2つにネットワークインターフェイスがそれぞれあることに注意する）
- 頭部カバーを閉めた状態で運用
    - 頭部のLAN端子への接続拒否設定は行えない
- 外部との通信はSSL等の暗号化されたプロトコル利用
    - 閉じられたローカルネットワーク、暗号化された通信路があれば、外部との接続を安全に行うことができるため

データの暗号化

パスワード・個人情報等、流失することで損害が発生する可能性があるデータをALMemoryに流さない。
止むを得ず流す場合は、お客様に流出リスク・影響を理解いただいた上で、暗号化処理を行う。さらに、暗号化を行う秘密鍵は別システムに保存するとセキュリティを高められる。（例：Pepper側にはトークン情報を保存し特定の環境（接続元、Pepper）が揃わないと秘密鍵が取得できないようにする）

ウイルス対策

Pepperに対応するアンチウイルスソフトはリリースされておらず、アンチウイルスソフトの適用ができない。従って、万が一ウイルスに感染したとしても影響範囲を限定させるために以下の対策を実施する。

- Pepper専用のWi-Fi回線を用意し、回線は原則Pepperと連携する外部システム、外部機器以外は接続できないようにする

確認事項（セキュリティ設計）		確認
1	意図しない第三者からPepperが接続するネットワークには接続できないようにすること（WPA2-PSKで保護すること）	
2	Pepperにセンシティブなデータ（パスワード・個人情報等）を保存する場合は暗号化を行うこと	

2.3.4.3. セキュリティ設計書レビュー・承認

セキュリティ上の方針についてお客様に説明し、承認を得る。

確認事項（セキュリティ設計）		確認
1	Pepperセキュリティ方針を説明し、お客様から承認を得ているか	

2.4. クラウド設計

2.4.1. プロセス概要

クラウド設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 外部システム（CMS（Content Management System）や外部クラウド）に関する設計を行う。
- Pepperに保持することができないデータの管理を整理する。
- クラウド設計書をお客様へ提示し、承認を得る。

2.4.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	要件定義書
3	作成資料	クラウド設計書
4	完了基準	お客様によるクラウド設計書の承認
5	後続プロセス	テスト計画

2.4.3. 作業の依存関係（クラウド設計）

2.4.4. 作業詳細

2.4.4.1. クラウド設計書作成

クラウド設計では外部システム（CMSや外部クラウド）の設計（データ保存）を行う。（基幹システムへの連携等、既存システムへデータ保存する場合は、後続プロセスである連携設計にて行う。）

Pepper本体ではセキュリティを担保することは難しく、また保存できるデータ容量に限りがある（ロボアプリ本体を含め100MB）。従って、データを扱う場合は、Pepper本体にデータは保存せず、外部システムに保存する設計方針を執る。

またデータを外部に保存するため、データ連携処理時の遅延対応の設計も行う。

〈Pepper本体に保存できないデータ例〉

- セキュリティが求められるデータ：認証情報、個人情報等
- 大容量データ：動画データ、音楽データ、高解像度画像データ等
- 変更回数が多いデータ：セリフ、画像データ等

設計にあたっては、以下を行う。

〈設計対象〉

- 稼動インフラの設計（選定・運用）
- 稼動ソフトウェアの設計（ソフトウェアスタック）
- フロントエンド
- バックエンド
- データ連携処理時の遅延対応
- クラウドでのデータ処理
- ログ
- データライフサイクル

稼動インフラの設計（運用）

稼動インフラの設計（運用）では、以下の資料を作成する。

〈作成資料〉

- ソフトウェアスタックまで記述したクラウド構成図（開発・テスト・本番環境ごとに作成）
    - 開発対象・管理対象の把握
    - 開発から運用への切替に利用（開発者・運用者間での認識の相違によるトラブル*稼動インフラの設計（選定）*
    - 稼動ソフトウェアの形態とコストを勘案し稼動インフラを決定する。
    - 回避）
- 利用ソフトウェアリスト、脆弱性情報取得頻度、パッチ適用手順
    - 脆弱性対応の迅速化

稼動ソフトウェアの設計

稼動ソフトウェアは、セキュリティを意識した以下の設計を行う。

〈考慮すべきこと〉

- パケットフィルタリング設定（IPアドレス、ポート、ルールによる回数制限）
- 基本的な攻撃方法の把握（XSS、SQLを含む各種インジェクション、DoS等）
- 脆弱性情報は、ソフトウェアの提供元が個別にWebサイトを通じてアナウンスを行っているが、全ての情報を把握することは困難であるため、脆弱性対策サイト[3]を利用し情報を収集した上で設計を行う

セキュリティを考慮した設計となっていない場合の影響は以下となる。

[3] https://www.ipa.go.jp/security/vuln/index.html を参照。

〈影響〉

- 追加コスト発生、スケジュール遅延の可能性
    - 後続プロセスである脆弱性診断の結果でセキュリティ担保のために大幅な設計変更を求められる
- セキュリティ事故（情報漏洩等）発生の可能性
    - 脆弱性診断を実施することでセキュリティ事後を確実に避けることはできないが、実施しない場合はセキュリティ事故の可能性は極めて高くなる。

フロントエンド・バックエンドの設計

稼動ソフトウェアはフロントエンドとバックエンドの設計を行う。またそれぞれの役割を明確に区別する。2つの役割を明確化するメリットは以下となる。

〈メリット〉

- 改修時の柔軟性の確保
    - フロントエンド・バックエンド、それぞれの接続先変更が容易
- 問題発生時の問題箇所の切り分けが容易

Pepperとクラウドの接続は、開発者ごとに別の接続方法で実装するのを防ぐため、具体的な手順とタイミングの定義を行う。

データ連携処理時の遅延対応

データ処理速度の高速化、回線速度の高速化、処理アルゴリズムの最適化、通信タイミング・回数の最適化、送信データ構造の最適化（差分データ送信等）を行う。

クラウドでのデータ処理

Pepperから受信したデータの処理、もしくはPepperへデータを送信する際にデータ処理が必要な場合は、その設計を行う。

ログ

ログの利用目的と取得方法を決定する。ログの主な利用目的は以下となる。

〈利用目的〉

種類		内容
1	エラーログ	問題発生時の範囲・問題・原因の特定
2	通常ログ	正常動作の確認（システムが正常に動作することを確認する）
3	監査ログ	認証の成功や失敗、システムの変更の記録（セキュリティ観点）

エラーログはロボアプリ品質チェックリストに記載の通り、ログを機体側に残すことは原則できない。必要である場合はクラウド側に保存する。その際の注意事項は以下となる。

〈注意事項〉

- 送信ログに個人情報が含まれる場合、エンドユーザの承諾が必要
    - ディスプレイに承諾ボタンを表示

- ロボアプリの設定画面でログ送信の有効化

データライフサイクル

クラウドには容量が大きいデータ・セキュリティデータが格納されているため、管理コスト削減・情報漏洩時の影響を軽減するために、データのアーカイブ、破棄を設計する。

設計する際に考慮する点は以下となる。

〈考慮点〉

- 対象データを利用する頻度とタイミングはいつか
  - 頻度が低い場合はアーカイブし、必要な時にだけ取り出せるようにする
- 対象データは、時間経過によって、利用にどのような変化があるか
  - 一定の時間経過によって利用されなくなるデータがある場合は破棄する

確認事項（クラウド設計）		確認
1	運用を想定する環境に合わせた設計になっているか 例：OS、ミドルウェア（バージョン等）	
2	クラウド構成図を作成しているか 例：クラウドの構成＋ソフトウェアスタックの構成	
3	クラウド側のセキュリティ設定は適切か	
4	フロントエンドとバックエンドの設計は明確か	
5	センシティブなデータ（パスワード・個人情報・流失することで損害が発生するリスクがある）を取り扱う場合、クラウドとの通信を暗号化しているか	
6	センシティブなデータを取り扱う場合、クラウドとの接続は適切な認証機構を設計しているか	
7	個人情報を取得する場合、データはクラウド側に保存する設計になっているか	
8	データライフサイクルに関する設計は行われているか 例：作成、保存、利用、アーカイブ、破棄等	
9	クラウドとの接続はシーケンス図あるいはプロトコルを記述しているか	
10	クラウドとの接続APIは詳細（プロトコル・データフォーマット含む）に設計されているか	
11	データ連携処理時の遅延対応を設計しているか	
12	エラーログの取得方法を設計しているか	
13	開発環境ないしテスト環境及び本番環境を想定した運用の仕組みを用意しているか	

2.4.4.2. クラウド設計書のレビュー・承認

クラウド設計書をお客様に提出し、承認をもらう。

確認事項（クラウド設計）		確認
1	クラウド障害発生時のリスクについて説明し理解を得られたか	
2	運用手法について確認が取れているか	
3	運用コストについて確認が取れているか（開発とは別に運用の月額コスト（稼動状況により金額が変化する等）	
4	開発・テスト・本番環境に対して運用責任が明確になっているか	

2.5. 連携設計

2.5.1. プロセス概要

連携設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 外部システム・外部機器と連携してロボアプリが動作する要件がある場合、連携設計（インフラ選定・連携方法の定義、データ構造の確認）を行う。
- 連携設計書をお客様へ提示し、承認を得る。

2.5.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	要件定義書
3	作成資料	連携設計書
4	完了基準	お客様による連携設計書の承認
5	後続プロセス	テスト計画

2.5.3. 作業の依存関係（連携設計）

2.5.4. 作業詳細

2.5.4.1. 連携設計書作成

連携するシステムとのインフラ選定、連携方法の定義、データ構造の確認を行う。データ構造の確認はスキーマレベルまで落とし込むことでバグの発生を抑止する。

〈スキーマ例〉

- XML：DTDあるいはXML Schema（xsd）

連携システム側で改修が必要な場合、各種ドキュメントの更新やシステム改修、テスト等が必要となるため、シナリオ設計完了次第、設計着手する。併せて、連携システムとのテストスケジュールの調整を行い、ロボアプリの開発スケジュールと整合性を取る。

連携設計は、円滑な運用を可能とするために、連携システムとの責任分解点、役割分担が重要となる。特に異常系処理は、連携システム、連携機器単位で発生する可能性があるエラーを出し、システム的なエラーと業務的なエラーに分類すると対応方針が明示しやすくなる。

〈システム的なエラー例〉

- 連携システムに繋がらなかった
- データが取得できなかった
- クーポンが発券されなかった
- ネットワーク接続ができなかった
- Pepperで異常値が出ていた

〈業務的なエラー例〉

- 空席がなかった
- 在庫がなかった
- 認証が一致しなかった

システム的なエラー、業務的なエラーいずれも発生した際の対応（業務フロー等）を用意し業務影響が出ないようにする。発生したエラーに対しては以下の対応を行う。

〈対応例〉

- システム的エラー
    - フェールセーフ対応する
      （例：Pepperのモーションを停止し、ディスプレイに「ただいま店員がまいります。少々お待ちください」と表示する）
    - フェールオーバ対応する
      （例：代替機を事前に用意し、業務を継続する。接続に失敗した場合は予備系システムへの接続先切り替えを行う）
    - 処理プロセスを遡りながらエラー箇所を特定する
- 業務的なエラー

○　エラーに対する業務フローを事前に用意する
（例：在庫がなかった→商品発注を行う）

上記を踏まえ、エラー原因の可能性がある対象に対して対応する担当者を決める。役割分担されたエラー対応方針例は以下となる。

〈エラー対応方針例〉
ロボアプリ：ヒアリングを行いオススメ商品を提示。在庫検索行い、在庫があった場合はクーポンを発券。

種類		エラー内容	対応方法	エラー対象	開発会社の対応	運用管理者の対応	現場オペレータの対応
1	システム的なエラー	クーポンが発券されなかった	モーションを停止し、ディスプレイに「ただいま店員がまいります。少々お待ちください」と表示する （処理プロセスを遡りながらエラー箇所を特定）	Pepper	・発券指示の処理は行ったか ・正しい認証方式で送信しているか ・CPU使用率が異常に上昇していないか ・メモリ不足になっていないか	-	-
				インタフェイス	-	・ネットワーク通信可能か	-
				プリンタ	-	-	・正しい認証方式で受信しているか ・電源は入っているか ・紙は残っているか ・紙詰まりしていないか
2	業務的なエラー	店頭在庫がゼロだった	Online Shopへ誘導 （事前に対応策を用意）	-	-	-	-
		店頭在庫、Online Shop共に在庫切れ	ディスプレイ・セリフ「お近くの店員をお呼び下さい」と表示・発話する（事前に対応策を用意）	-	-	-	・該当商品の取り寄せ処理を行う

確認事項（連携設計）		確認
1	連携システムとの通信方式・メッセージデータの構造を記述しているか	
2	連携障害発生時の設計を盛り込んでいるか	
3	連携システムとの役割分担を明確にしているか	
4	連携システムの改修が伴う場合、スケジュールの調整を行い、開発スケジュールと整合性を取っているか	

2.5.4.2. 連携設計書のレビュー・承認

連携設計書をお客様に提出し、承認を得る。

2.6. 展開計画

2.6.1. プロセス概要

展開設計では、主に以下の作業を想定している。

〈主な作業〉

- ロボアプリ機能を段階的にリリースする場合、または複数台のPepperを段階的にリリースする場合、その展開スケジュールを作成する。
- 展開計画書をお客様へ提示し、承認を得る。

2.6.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	要件定義書
3	作成資料	展開計画書
4	完了基準	お客様による展開計画書の承認
5	後続プロセス	テスト計画

2.6.3. 作業の依存関係（展開計画）

2.6.4. 作業詳細

2.6.4.1. 展開計画書作成

機能を段階的にリリースする場合、または複数台のPepperを段階的にリリースする場合、展開計画書を作成する。作成は、開発会社・お客様で話し合い担当を決める。展開計画書に含まれる項目は以下となる。

〈展開計画書の項目例〉

- 展開マスタスケジュール
    - 申請から納品・設定（必要に応じて現地指導）まで
    - 周辺機器（Wi-Fi環境等）の構築・設定含む
    - Pepperの納品は時間指定のサービスを行っていないため、現地で初期設定のサポート、レクチャー等を行う予定の場合は、前日までに納品しておくことが望ましい。保管場所を確保しておく等の調整が必要となるため、計画に盛り込む
- 展開場所、台数
- キッティング・配送
    - 作業項目・作業フロー
        - Pepper納品後の初回設定、ロボアプリ配信・ダウンロードを代行するか否かを含む
    - 実施タイミング
    - キッティング作業前の準備事項
        - ID管理、機体管理

確認事項（展開計画）		確認
1	調達リードタイムを考慮した展開マスタスケジュールを作成しているか	
2	展開場所の記載はあるか	
3	展開台数の記載はあるか	
4	関連システムのライフサイクルに合わせているか（関連システムがバージョンアップした場合、対応が必要）	
5	導入方法（Choregrapheで導入もしくはクラウドを利用してロボアプリ配布）が確定しているか	
6	キッティングのフローと作業項目を定義しているか	
7	キッティング・配送等のタイミングを明確化しているか	
8	キッティング作業前に必要な準備事項を定義しているか 例：ID管理、機体管理、移動経路、関係者との調整等	

2.6.4.2. 展開計画書レビュー・承認

展開計画書をお客様に提出し、承認を得る。

2.7. 運用設計

2.7.1. プロセス概要

運用設計プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- Pepper導入後はお客様側のサポート体制を定義し、円滑なPepper利用・トラブルシューティングが可能な体制を構築する。
- 運用設計書をお客様へ提示し、承認を得る。

2.7.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	要件定義書
3	作成資料	運用設計書
4	完了基準	お客様による運用設計書の承認
5	後続プロセス	テスト計画

2.7.3. 作業の依存関係（運用設計）

2.7.4. 作業詳細

2.7.4.1. 運用設計書作成

運用設計では以下の項目を設計する。（お客様が実施する運用設計を行う。開発会社の運用設計は後続プロセス「メンテナンス準備」で行う。）

〈設計項目〉

項目		内容
1	運用体制構築	役割、責任、主なタスクを記載する。

2	運用業務一覧	平常時運用、問い合わせ・不具合時対応、故障交換時対応、事故対応（例：Pepperのモーションでエンドユーザに危害を与えてしまった）等それぞれの運用業務に対して、担当者を整理する。
3	各運用フロー	運用業務一覧で整理した業務について、業務フローを作成する。
4	運用メンバ教育実施方針	教育実施方法や対象者、実施スケジュールを整理する。
5	運用条件	Pepperの保管方法等を整理する。

一般的な運用設計に加え、Pepper固有の以下運用条件を整理する。

〈運用条件〉

- Pepperの保管方法[4]
- 設置場所までの移動ルート
- 設置場所までの移動ルール・方法
    - 例：台車に載せて運ぶ、台車には2人で持ち上げて載せる
- Pepperの動作フローに基づく、設置場所に必要な準備物の定義
    - 例：什器、（注意文の書かれた）ポスター、カート侵入防止枠等
- Pepper設置時間中に行うべきオペレーション
    - 例：充電を抜かない、フラッパーゲートを上げておく、再起動の実施、レストのための予備Pepperのローテーション等
- Pepperの梱包箱の保管場所
- Pepperの回線確保
    - 例：Pepper専用のモバイルWi-Fi（スリープ解除設定、電源確保、設置場所、契約）

確認事項（運用設計）		確認
1	役割・責任・主なタスクが整理された運用体制となっているか	
2	運用業務に抜け漏れはないか	
3	運用体制と運用業務一覧の担当者の整合性が取れているか	
4	業務実施にあたり障壁となりそうな問題・課題の洗い出し、及び解決策の検討・実施を行っているか（実現可能な運用フローに落とし込んだか）	
5	運用メンバに必要なスキルの洗い出しを出来ているか	
6	運用スキル習得に向けて、具体的かつ実現可能な教育方法・スケジュールを組んでいるか	

2.7.4.2. レビュー・承認

運用設計書をお客様に提出し、承認を得る。

[4] http://www.softbank.jp/robot/consumer/support/start/ を参照

2.8. 非機能要件

2.8.1. プロセス概要

非機能要件プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- Pepperの稼動時間、ログ出力内容、前提事項、設置場所、免責事項等の非機能要件の設計を行う。
- インタラクションラベル（シナリオ設計と連携）で集計する情報の設計を行う。
- 非機能要件設計書をお客様へ提示し、承認を得る。

2.8.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	要件定義書
3	作成資料	非機能要件設計書
4	完了基準	お客様による非機能要件設計書の承認
5	後続プロセス	テスト計画

2.8.3. 作業の依存関係（非機能要件）

2.8.4. 作業詳細

2.8.4.1. 非機能要件設計書作成

想定外あるいは高負荷状態での利用を防ぎ、安定的な運用のために作成する。

非機能要件が不明確な場合、後続プロセスであるテスト計画・開発・運用でトラブルの原因となるため、お客様と共通認識を持つことが必要となる。（具体的な非機能要件は付属資料「非機能要件」を参照のこと）

確認事項（非機能要件）	確

		認
1	非機能要件の項目が全て網羅され、各項目にお客様とギャップ（認識のズレ）がないか	

2.8.4.2. 非機能要件設計書のレビュー・承認

非機能要件設計書をお客様に提出し、承認を得る。

2.9. テスト計画

2.9.1. プロセス概要

テスト計画プロセスでは、主に以下の作業を想定している。

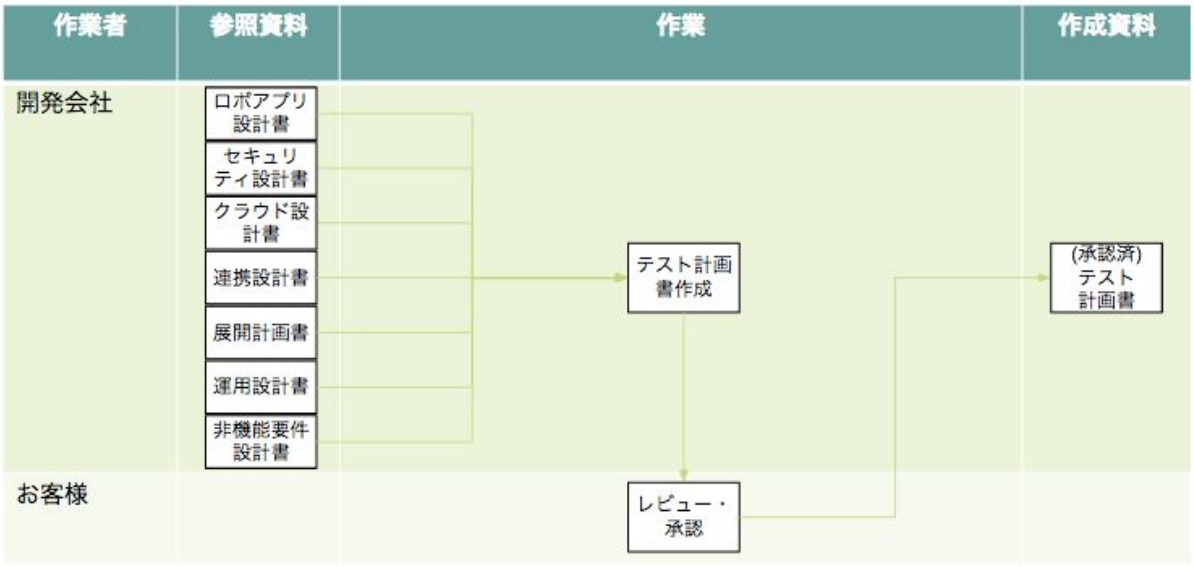
〈主な作業〉

- 各種テストの目的、テスト大項目、テスト対象、テスト方法、テスト環境、スケジュール、体制・役割分担、テストデータの準備方法等の計画を策定する。
- ロボアプリ品質チェックリスト記載のテストを確認し、実施に向けて計画に盛り込む。
- テスト計画書をお客様へ提示し、承認を得る。

2.9.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	ロボアプリ設計書、セキュリティ設計書、クラウド設計書、連携設計書 展開計画書、運用設計書、非機能要件設計書
3	作成資料	テスト計画書
4	完了基準	お客様によるテスト計画書の承認
5	後続プロセス	開発

2.9.3. 作業の依存関係（テスト計画）



2.9.4. 作業詳細

2.9.4.1. テスト計画書作成

テストごとに目的、項目、対象、テスト方法、環境、実施スケジュール、体制・役割分担、テストデータの準備方法等の計画を策定する。

テスト計画・項目作成にあたって、各プロセス設計者・開発者とは別の者がテスト実施者として作成すると、設計や開発を意識しないため、抜け漏れ発生を回避できテスト品質の向上が見込める。設計は具体的に以下テストごとに作成する。設計にあたっては、ロボアプリ品質チェックリスト記載のテストを確認し、計画に盛り込む。

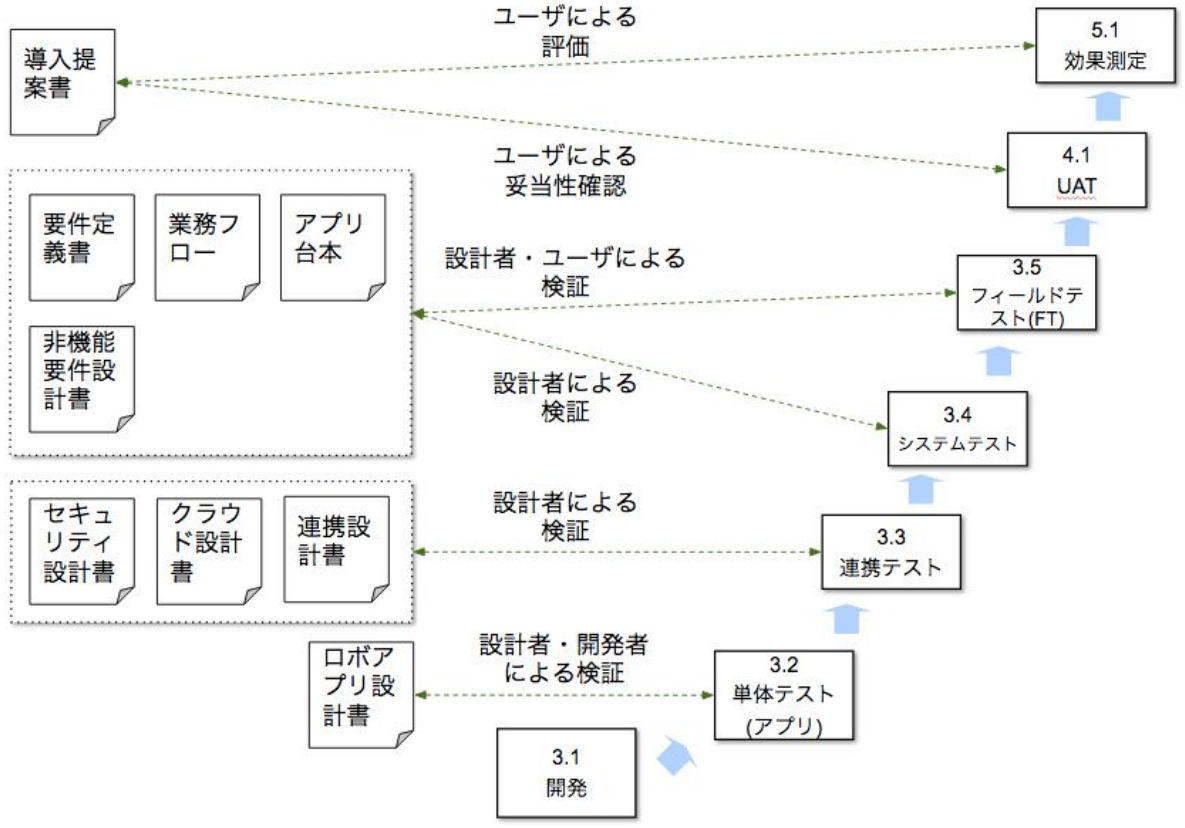
〈テストの定義〉

テスト名		定義
1	単体テスト（ロボアプリ）	ロボアプリ単体テストのことをいう。本ドキュメントでの単体テストという場合、ルーチンに置ける単体テストやモジュールにおける単体テストという意味では用いない。ルーチン、モジュール、後述するビヘイビアが実装されたロボアプリの単体テストのことである。 ①ビヘイビア単体での動作確認と、②ビヘイビア間（またはお仕事かんたん生成との間）での連携動作を確認する。
2	連携テスト	ロボアプリと、ロボアプリ関連システムが正常に機能することを確認する。また、外部システム、外部機器（CMSや外部クラウド、その他）等と連携がある場合は、外部システム、外部機器等と、それぞれ個別に接続した状態で行うテストのことである。

3	システムテスト	実際の本番環境（システム・設置環境）に近い環境を用い、システムが正常に機能するか、業務フローが機能するか、確認するテストのことである。 連携する外部システム、外部機器がある場合は、それらも含めて全体として要求を満たしているかの観点で行うと共にパフォーマンス・応答性の確認も行われる。
4	フィールドテスト	実際の本番環境（システム・設置環境）を用い、システムが正常に機能するか、業務フローが機能するか、確認するテストのことである。 お客様にロボアプリを確認してもらうため、一番重要なテストとなる。 連携する外部システム、外部機器がある場合は、それらも含めてシステムが全体として要求を満たしているかの観点で行うと共にパフォーマンス・応答性の確認も行われる。 システムテストとの差異は、①本番環境である、②開発会社が主導でありながら現場オペレータもテストに参加・実施する、の2つとなるため、システムテストで確認できなかった内容が含まれる形となる。 また、実際のエンドユーザになりきったモンキーテスト（思いついたことを手当たり次第に行うテスト）を実施し、通常のテストでは検出できないシステムの不具合がないかチェックする。
5	UAT（User Acceptance Test）	一連のテストプロセスが完了し、開発したロボアプリを含めたシステムが本来の目的通りに機能するか、お客様が確認する受入テストのことである。 フィールドテストとの差異は、お客様自身が主導してテストを作成・実施する点である。

プロセスの対応関係

企画・要件定義、設計フェーズの作成資料と、各テストの関係は以下となる。



単体テスト（ロボアプリ）

ロボアプリ設計書に対する検証は、単体テスト（ロボアプリ）で実施する。

連携テスト

セキュリティ設計書、クラウド設計書、連携設計書に対する検証は、連携テストで実施する。以下の制約やリスク等を事前に洗い出し、対応策を決め計画を策定し実施する。

〈制約、リスクの例〉

- テスト開始可能時期が、連携する外部システムの改修スケジュールに依存する
- 通常利用に影響を与えないようテスト可能な時間帯が制限される
- 外部システムのテスト環境を用意する必要がある（24時間365日外部システムが稼動している場合）
- 本番データとは別のテストデータの準備が必要となる（本番データを利用する場合、データのマスキング有無、テストデータ消去方法を決める）

また、テストを本番環境で実施する場合は、トラブルが発生した場合の対応（対処方法）及びテスト終了後の回復方法（テストデータ消去、設定変更の戻し等）を検討、決定しておく。

システムテスト

全てのシステムを結合させ、本番とほぼ同様の環境で行うテストのことをいう。要件定義書、業務フロー、アプリ台本、非機能要件設計書の検証プロセスとなる。

フィールドテスト

全てのシステムを結合させ、本番環境で行うテストのことをいう。システムテストとの大きな差は「本番環境そのもの」で行うことであり、また現場オペレータ自身もテストに加わることである。現場オペレータは主に業務要件に関わるテストを担当する。併せてテスト担当者は、実際のエンドユーザになりきったモンキーテスト（思いついたことを手当たり次第に行うテスト）も実施し、通常のテストでは検出できないシステムの不具合がないかチェックする。運用開始後のトラブル回避のために以下を考慮し、テスト期間・負荷テストを計画する。

〈考慮点〉

- エンドユーザの変化（日中は子供、夜間は社会人等、エンドユーザ属性が変わる）
- 環境の変化（時間帯・平日・休日・繁忙期・閑散期での人の多さ・騒音・光量等）

テスト用Pepper・機材の保管場所確保が必要となるため、実施場所が確定次第、テスト用Pepper・機材の納品・返却タイミングの調整を行う（お客様が既にPepper導入済の場合は、使用中のPepper・機材の利用調整を行う）。また、連携テストと同様、テスト実施中に本番環境でトラブルが発生した場合の対応（対処方法）及びテスト終了後の回復方法（テストデータ消去、設定変更の戻し等）を検討し、決定しておく。

UAT（User Acceptance Test）

導入提案書に対する妥当性確認はUATで実施する。フィールドテストとの差異はお客様自身が主導してテストを作成・実施する点である。UATは基本としてフィールドテストまで一通り完了した段階で行われる。

効果測定

導入提案書に対する検証は効果測定で実施する。導入提案書記載のKGI・KPIが達成できたどうか確認する。

目的の設定

目的を設定する際は、ロボアプリの品質目標を意識できる内容で記載する。品質目標を決めることで、テスト項目の品質チェックが明確となりテスト品質の向上が見込まれる。

〈テストの目的例〉

- 単体テスト：Pepperの発話、ディスプレイ表示、モーションがロボアプリ設計書と相違ないことを確認する
- 連携テスト：構築したCMSがクラウド設計書と相違ないことを確認する
- フィールドテスト：XXXX機能は、正常処理は一部条件で確認、異常・例外処理は、全条件で確認する

不合格の対応方針、管理方法の決定

全体スケジュールに影響を与えるため、テスト方法の策定では、テストで不合格になった場合の対応方針、管理方法を記述する。不合格になった内容は、テスト仕様書に記載する結果とは別に一元管理して対応する。

〈対応例〉

- バグ管理表に記載し、管理者がお客様と都度、対応を検討する

- チケット管理ツール等を利用し、管理者・担当者がお客様と定例会議で、対応を決定する

関連資料の整理

具体的なテスト項目・テスト内容を作成する際に参照する作成物（定義書・仕様書）の整理を行い、計画書に関連資料として明示しておく。

〈記載メリット〉
- 設計書・定義書に変更があった場合、テストの抜け漏れを防ぐ
- レビュー者・承認者がテスト計画・テスト内容の品質を評価する際の材料とする
- テスト計画の変更、追加が発生した際に責任の所在が明確化できる（お客様に設計・開発・テストの関係を理解してもらい、実施の変更やテスト追加の話をしやすくする）

設計漏れの対応（テスト項目作成）

テスト項目の作成で、参照する設計書に以下の漏れがあった場合、対応（すぐに検討及び設計書へ反映するのか、変更管理として別途処理するのか等）が必要となる。

〈設計漏れの例〉
- 設計書に記載されていない内容がある
    - 例：エラー処理、例外処理が抜けている
- 曖昧な設計がある。
    - 例：「データ入力させる」……漢字カナ入力は受付可能なのか？　半角英数字の場合はどうなるのか？　文字数の制限はいくつなのか？
    - 例：「AとBとCを同時に実行させる」……平行で処理を行うと厳密には同時に実行されることはない。

設計・開発の品質向上に繋げるため、テスト仕様書作成者は、設計書からコピー&ペーストでテスト項目を作成するのではなく「抜け漏れがないか」、「曖昧なところはないか」という視点でテスト項目を作成する。
また、設計漏れが発生した場合にどのような対応を行うのか対応方針、管理方法等を決定しておく。

〈対応例〉
- 変更管理書に記載する
- 週次で変更管理会議を開催し、対応方法を決定する

テストにおけるエビデンス取得方式の合意

テストの実施結果はエビデンス（証拠ないし結果）として、開発会社・お客様双方が理解可能な形で保存する必要がある。ロボアプリ開発におけるエビデンスの取得方法は、以下の方式を想定する。

〈方式〉
- 期待どおりに動作した、しなかったの結果（合格・不合格）をテスト仕様書に記載
- システム側から返された値の結果をテスト仕様書に記載
- ロボアプリのディスプレイの画像

- ロボアプリのモーションを撮影した動画
- ロボアプリのモーションを複数人でチェックした結果をテスト仕様書に記載
- ロボアプリの音声の録画・録音
- ロボアプリの音声結果をテスト仕様書に記載

エビデンスの取得方式は、開発会社・お客様双方で合意が必要となる。
特にロボアプリのモーションを複数人でチェックする方法等、形として残っていない方式を採用する場合は、実施方法や確認者の選定（ダブルチェックにお客様を入れるか）等をお客様に説明し、認識の相違がないことを確認した上で合意する必要がある。

また、エビデンスとして有効となるには以下の情報を合わせて取得しなければならない。

〈取得情報〉
- 対象のソフトウェア情報（ロボアプリ・連携システムのバージョン）
- テストの実施方法
- テストを行った実施者
- テストの実施日
- テスト実施の回数

注意点として、エビデンスを取得してもハード障害やモーターホット等の不具合は発生することをお客様と共有する。

確認事項（テスト計画）		確認
1	テストの定義と目的が設定されているか	
2	テスト不合格の対応方針、管理方法が設定されているか	
3	エビデンス取得方式が設定されているか	

2.9.4.2. レビュー・承認

テスト計画書をお客様に提出し、承認を得る。開発・テストフェーズへ進み、設計がお客様責任で変更された結果、設計フェーズに立ち戻った場合は別途追加見積となることを、お客様と合意する。

確認事項（テスト計画）		確認
1	テストの定義と目的の合意が得られているか	
2	テスト不合格の対応方針、管理方法の合意が得られているか	
3	エビデンス取得方式について合意が得られているか	
4	単体テストの各調整事項（テスト実施期間等）について合意が得られているか	
5	連携テストの各調整事項（テスト実施期間、接続方法、テストデータ準備方法等）について合意が得られているか	

6	システムテストの各調整事項（テスト実施期間、設置・保管場所、設置環境等）について合意が得られているか	
7	フィールドテストの各調整事項（テスト実施期間、設置・保管場所、設置環境、テストデータ準備方法等）について合意が得られているか	
8	UATの各調整事項（テスト実施期間等）について合意が得られているか	

3. 開発・テスト

開発・テストフェーズは、ロボアプリ設計書を基に開発を行い、設計フェーズで計画したテスト計画書に従いテストを実施するフェーズである。開発プロセスではロボアプリ設計書を基に、ロボアプリ開発を行う。

開発後テストプロセスに入り、単体テスト、連携テスト、システムテスト、フィールドテストを計画書に従い実施する。テスト結果を開発者にフィードバックし、設計書記載の要件を満たすよう改善を繰り返す。

本フェーズは「3.1 開発」「3.2 単体テスト（ロボアプリ）」「3.3 連携テスト」「3.4 システムテスト」「3.5 フィールドテスト」の5プロセスとなっている。

3.1. 開発

3.1.1. プロセス概要

開発プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- ロボアプリ設計書に基に開発を行う。
- 開発内容がロボアプリ品質チェックリストに準拠していることを確認する。
- 開発会社の管理者がソースコードレビューを了承する。

3.1.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	ロボアプリ品質チェックリスト、ロボアプリ設計書
3	作成資料	ソースコード、実行ファイル（pkgファイル）
4	完了基準	開発会社内でのソースコードの了承
5	後続プロセス	単体テスト（ロボアプリ）

3.1.3. 作業の依存関係（開発）

3.1.4. 作業詳細

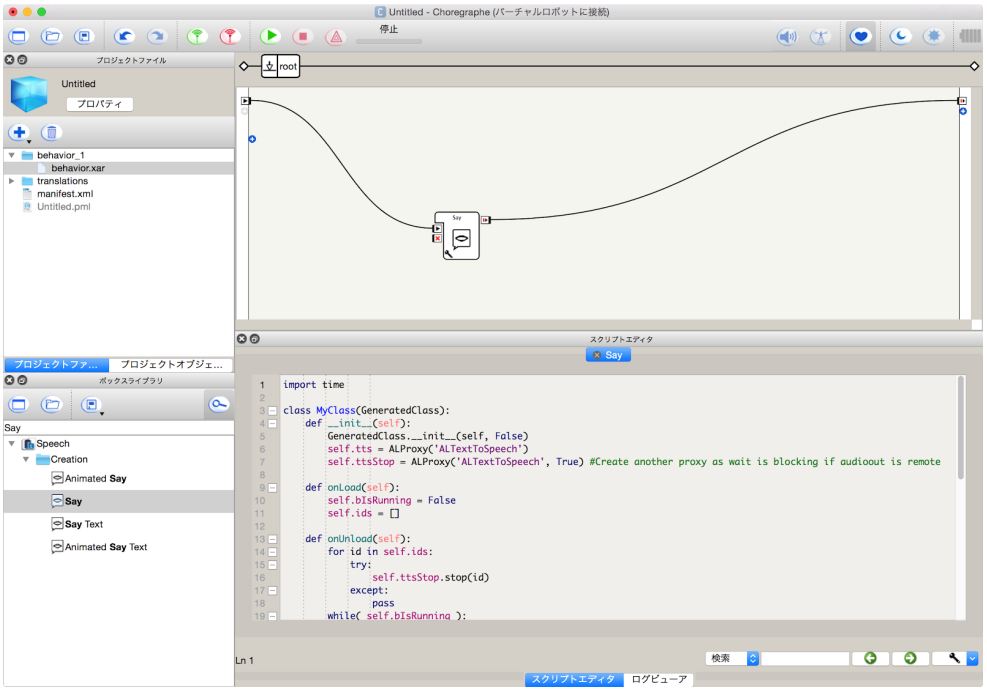
3.1.4.1. 開発

　ロボアプリ設計書を基にロボアプリ開発を行う。開発の効率化のため、命名規則・コーディング規約等各種規則・規約やロボアプリ品質チェックリストの内容を確認する。開発後は、その内容がロボアプリ品質チェックリストに準拠していることを確認する。
　Choregrapheの役割とロボアプリの構成を理解しておくとロボアプリ開発プロセスのイメージを把握しやすいため、以下に概要を記述する。

Choregrapheとロボアプリ

　ChoregrapheはNAOqi OSで動作するロボアプリを開発するGUIツールである。開発はChoregrapheにボックスと呼ばれるパーツ（コード片）を配置し結線して行う。

Choregrapheのイメージ



ロボアプリは以下の階層構造を持つ。

〈階層構造〉

- ロボアプリ
  ├アイコン
  ├マニフェスト
  ├ビヘイビア
  │ └ボックス
  ├サービス（オプション）
  ├ライブラリ（オプション）
  └リソース（オプション）

アイコン

アイコンはロボアプリの顔にあたる重要な構成要素である。従って、ロボアプリに期待される動作イメージを表すようにする。他のロボアプリと一貫されたデザインを維持するために、六角形のテンプレート[5]に従ってデザインする。デザインされたアイコンはロボアプリ配信管理に表示される。

マニフェスト

マニフェストはロボアプリの概要を宣言するファイル（manifest.xml）である。設計段階で決定したアプリ情報を記載する。記載にあたっては実行情報の宣言と配信システムの制御情報も同時に含むため注意する。

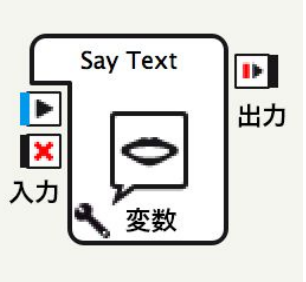
〈マニフェストが持つ情報〉

- ビヘイビアの宣言
- サービスの宣言
- ロボアプリの起動条件
- ロボアプリのバージョン情報
- ロボアプリの名称
- ロボアプリの説明

ビヘイビア

NAOqi OSが処理を行うロボアプリ本体である。ビヘイビアという名前の通り、ロボアプリのふるまいがプログラムされている。XMLで記述されたファイルbehavior.xar内にプログラムが格納されており、ビヘイビアの中にはボックスという複数のコード片が含まれる。作成は、XMLエディタでなくChoregrapheにて行う。

ボックス



ボックスはビヘイビアを構成する処理単位である。基本的なボックスは、入力と出力及び変数を持ち関数のように機能する。ロジックの構築はボックスを結線して行う。

ボックスを利用する場合の注意点は以下となる。

〈注意点〉

[5] アプリアイコン http://www.softbank.jp/robot/biz/support/document/icons/

- Choregraphe付属の標準ボックスを利用し、標準ボックスに修正を行った場合は、ボックスの名称を変更する。

ビヘイビア中で中心的な機能を担うボックスでは以下のような表をつくり、ボックスの役割を明確にし管理する。

〈ボックスの役割例〉

項目		記述例
1	ボックス名	Get CMS User Access Token
2	階層	/root/CMS Connect/
3	役割	CMS接続用のユーザアクセストークンを取得する
4	変数（ボックス）	CMS Address：CMSのアドレス（型：String）
5	入力	onInput_onStart():
6	出力	onStopped(result): result：結果 none:失敗 String: Token（型：Dynamic）

ボックスの役割を管理するメリット、は以下となる。

〈メリット〉
- 類似ボックスの重複開発の抑止
- 開発者間の情報共有（例：開発段階で明らかになったタイミングを調整するためのボックス等）

サービス

サービスはビヘイビアとは独立し、NAOqi OSモジュールと同様に動作するプログラムである。サービスの利用メリットは以下となる。

〈メリット〉
- NAOqi OSの一部として振舞い、ボックスのライフサイクルの影響を受けない
- NAOqi OSとは別スレッドで処理される（メインとなる処理に基本的には負荷をかけない）
- 開発はPythonもしくはC++等で行え、Choregrapheによる制限を受けない
- ボックスのライフサイクルとは独立しており、Choregrapheでは難しい本格的なオブジェクト指向開発（処理のモジュール化等）が可能となる

ただし、システムに影響を与える可能性が高いため、以下の点に注意が必要となる。

〈注意点〉
- 標準のモジュールに干渉しない実装とすること
- ネームスペースの衝突を防ぐため、独自モジュール開発会社名を示す接頭辞をつけること

例：先頭に開発会社名を示す大文字アルファベット2～3文字<VendorNamePrefix>＋モジュールの役割が明確になる名前<ModuleRole>
<ModuleName> ::= <VendorNamePrefix><ModuleRole>
→SBRDataBaseConnector

- アプリ終了後は独自モジュールをアンロードすること

ライブラリ

ロボアプリの開発を効果的に行うため、公開されているライブラリ[6]を利用する。利用ライブラリ情報はドキュメンテーションを作成し、開発者間で共有する。また、利用上の注意点は以下となる。

〈注意点〉

- ライブラリはバージョンによる動作の変更が伴うため、バージョン管理を徹底する
- OSSライブラリのライセンスを確認し、ライセンスに基づいた利用を徹底する

リソース

リソースとは、ロボアプリ内マニフェスト・ビヘイビア・ライブラリ・サービス等に該当しないその他構成物をいう。リソースはフォーマットの制限を基本的に受けないため、有効活用することでコンテンツとプログラムの分離等が行える。

〈主なリソース〉

- 表示をコントロールするHTMLやCSS
- ロボアプリ内で解釈されるコンテンツ

リソースとして内部プログラムから解釈実行可能なドメイン固有言語（DSL）によるプログラムが可能であるが、ロボアプリでは基本的に禁止されている。ただし、コンテンツとして動作するチューリング完全でないDSLであれば問題ない。

C++でのライブラリ・サービスの作成

以下のシーンではC++による開発を検討する。

〈利用シーン〉

- 高速動作が求められる場合（Pythonコードでは速度要求が満たせない）
- ソースコードが秘匿が求められる場合

〈注意点〉

- C++でライブラリ・サービスを開発する場合、NAOqi OSのカーネルバージョン、システムライブラリにバージョン差異があると稼動しない（もしくはカーネルクラッシュが発生する可能性がある）
- NAOqi OSバージョンアップ時は動作チェックが必要

効果的なコーディング

Choregrapheは機能ごとに分割されたボックスを結線してコーディングを行う。ただし、ボックスが増えるほどマシンリソースを圧迫しプログラムの動作が不安定になる。そのた

[6] https://pypi.python.org/pypi にて一般的に公開されている。

め、Pythonボックスを使用し、一つのボックスで複数のボックスと同じ働きをするようにコーディングを行い、ボックスの集約化を行う。メンテナンス性の向上し、動作速度の高速化が期待できる。
　また、共通して利用するプログラムであればChoregraphe上のボックスではなく、外部ライブラリやNAOqi OS上のサービスを利用する[7]。

APIの利用

　標準で用意されているボックスでは、ほぼ一つのAPI[8]の呼び出しを目的としていることが多い。従って、ボックスを結線してプログラムを行う形となり、結果としてボックスが増加する傾向にある。ボックス削減による処理の安定化及び保守性の向上を見込み、機能を集約したPythonボックスによるAPI呼び出しを行う。
　Pythonボックスでは、一つのボックスの中に複数のAPI呼び出しが可能となる。ただし、BizPackの提供する「サービスの動作・挙動」に影響を与えないように注意する。

APIの種類

　NAOqi APIには標準APIに加えて、非推奨・廃止予定（Deprecated）のAPIがある。
　Deprecated APIは将来のNAOqi OSにおいて廃止される予定であり互換性維持のために一定期間残されている。同指定がついているAPIは新しいAPIによる代替実装を行うべきである。Deprecated APIはSDKドキュメントで確認できる。

Pythonによる開発とコーデング規約

　Pythonに関する開発は以下のサイトを参照する。
　Python公式サイト（英語：https://docs.python.org/devguide/）
　コーディングスタイルは、Python公式サイト記載のPEP8コーディング規約を準拠する。
　Python公式サイト（英語：https://www.python.org/dev/peps/pep-0008/）[9]
　規約準拠により、ソースコードの可読性が高まり、共同作業者・レビュー者の作業時間が短縮が測れ、レビュー者が問題を発見しやすくなり運用保守性が高まる。

ルールの標準化

　開発品質向上のために、会社独自の規約を制定する。規約に記載する項目は以下となる。

〈項目例〉
- 禁止事項：使用禁止機能やクラス、ライブラリ等。禁止理由を記載。
- 制限事項：使用を推奨しない機能やクラス、ライブラリ等。制限とする理由、条件を記載。
- 推奨事項：使用を推奨する機能やクラス、ライブラリ等。似たようなクラスがあった場合、どちらを推奨するか等を理由や条件と共に記載。

　ただし、規約の運用は以下点に注意する。

〈注意点〉
- 技術的に古くなっていないか

[7] インスタンスのライフサイクル（生成・破棄）の都合で同じ処理のボックスを複数配置している。このような場合に、サービスを利用しモジュール化すれば複数の同一ボックス配置の必要性はない。
[8] http://doc.aldebaran.com/ を参照。
[9] 日本語訳：http://pep8-ja.readthedocs.io/ja/latest/

    - 過去プロジェクトで使用したものをそのまま流用しない
- 開発メンバのスキル、プロジェクトの要件規模に合っているか
    - 保守性と開発の生産性のバランスを取る
    - 人数が多くスキルがバラバラの場合は、禁止・制限・推奨事項は細かく記載する。高スキル少人数メンバの場合は、公式サイト記載内容から抜粋したものにとどめる。高い機密性・完全性・可用性が求められる場合は、細かく規定する等

統合開発環境の利用

Choregrapheはロボアプリの開発環境であるが、ディスプレイUI・ライブラリ・サービスの開発は行うことができない。これらの構成要素は、テキストエディタ等で開発できるが、必要に応じて統合開発環境（IDE）を利用する。また、統合開発環境（IDE）の利用は以下のメリットがある。

〈メリット〉
- メソッド名の補完
- インデントやスペースなどの自動整形支援機能
- 文法の自動チェック

確認事項（開発）		確認
1	開発内容がロボアプリ品質チェックリストに準拠しているか	
2	Choregraheのフローダイアグラムに依存しすぎた開発手法を取っていないか（サービスやライブラリを使用しているか）	
3	開発したボックス情報の共有はされているか	
4	C++でライブラリやサービスを作成する場合、適合システムバージョンの整合性を確認したか （NAOqi OSのカーネルバージョン、システムライブラリにバージョン差異があると稼動しないことがあるため）	
5	Pepper本体は、Pepper for Bizの契約に伴い、弊社から提供した機体を使用したか	
6	NAOqiは、最新のリリースVersionを使用したか （Pepper管理画面の設定から最新かどうか確認する）	
7	Pepper for Bizの設定アプリ上で、全てのアプリケーションがインストール済みであるか	
8	開発は、最新のリリースVersionのSDK（Choregraphe）で行ったか	
9	ロボアプリの登録は、pkgファイルを使用したか （pkgファイルは、Choregrapheメニュー［ファイル］-［パッケージをビルド…］より作成できます）	
10	ロボアプリ容量は、本体及びダウンロードコンテンツを含め100MB以内とし、また軽量化に努めたか （容量が大きい場合、ロボアプリ起動に時間がかかりユーザーストレスとなる場合がある）	

11	アプリケーションIDは公開方法に応じ以下のルールに従っているか ロボアプリマーケット for Biz　：　biz_market_<会社名>_<ロボアプリ名> ロボアプリ限定公開　：　biz_<会社名>_<ロボアプリ名> ※半角英数字及び-（ハイフン）_（アンダーバー）が使用可能 例：biz_market_softbank_oshigoto-launcher biz_softbank_oshigoto-launcher	
12	登録済ロボアプリをバージョンアップする場合、同一のアプリケーションIDを使用しているか	
13	ロボアプリの概要は、対応する各言語でそれぞれ同様の内容を盛り込んでいるか	
14	対象NAOqiバージョンは、最小と最大の記載があるか 最小・最大共に動作確認できているバージョンを2~3桁で記載すること。 ※例：NAOqi2.5.5.5で動作確認している場合は「2.5.5」もしくは「2.5」を指定する。 ※本ガイドラインv1.0.0では、最大バージョンに"2"を指定するという表記があったが、2018年8月から最小・最大共に1桁の表記は禁止とする。	
15	ユーザーのリクエストより開始は、チェックを外しているか	

3.1.4.2. レビュー・了承

開発会社側で開発成果物（ソースコード等）のレビューを行い次の工程に引き継ぐ。開発者本人によるソースコードレビューのみの場合、開発者自身の思い込みによる問題を発見することが難しいため、別の担当者によるダブルチェックを行う。

またソースコードレビューでは以下内容を記録に残しておく。

〈記録内容〉

- 実施日（どの時点のコードであるかわかるようにするため）
- 実施者
- 対象範囲
- 指摘事項

レビューは指摘事項に対しての修正案を考えるのではなく、できるだけ多くの問題を指摘することに集中する。指摘事項は、対応が完了するまで管理する。

確認事項（開発）		確認
1	開発者とは別の担当者がソースコードレビューを行ったか	
2	開発内容がロボアプリ品質チェックリストに準拠しているか	
3	命名規則、コーディング規約等規則・規約に準拠しているか	
4	品質の特性（機能性・信頼性・使用性・効率性・保守性・移植性）を担保できているか	
5	可読性があるか（コメントが適切である。無駄な改行・コードの記述がない。変数や定数の表現・設定に一貫性がある等）	
6	マニフェストを適切に記述しているか	

7	レビューの指摘事項の対応が完了しているか	

3.2. 単体テスト（ロボアプリ）

3.2.1. プロセス概要

単体テスト（ロボアプリ）プロセスでは、主に以下の作業を想定している。

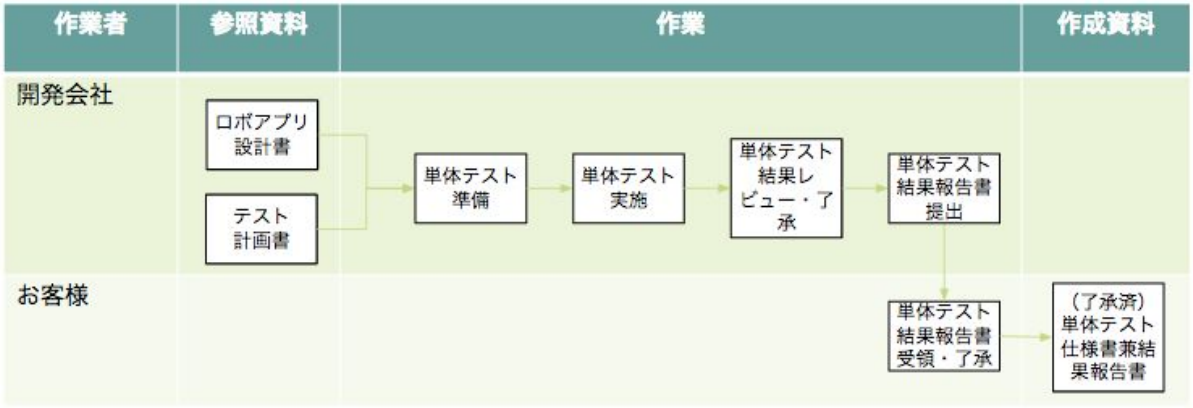
〈主な作業〉

- ロボアプリ設計書、テスト計画書を基にロボアプリ単体テスト仕様書を作成し、テストを実施する。
- テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。
- 開発会社内了承済みのテスト結果を報告書としてお客様へ提出する。

3.2.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	ロボアプリ設計書、テスト計画書
3	作成資料	単体テスト仕様書兼結果報告書
4	完了基準	お客様による単体テスト仕様書兼結果報告書の了承
5	後続プロセス	連携テスト、システムテスト

3.2.3. 作業の依存関係（単体テスト（ロボアプリ））



3.2.4. 作業詳細

3.2.4.1. 単体テスト準備

テスト計画書、ロボアプリ設計書を基にロボアプリ単体テスト仕様書を作成する。

また、必要に応じてテスト用のモックを配備する等外部連携用の準備を行う。

また通常の動作確認に関わるテスト項目に加え、以下を対象としてテスト項目を作成する。

〈その他テスト対象とその目的〉

テスト対象		テスト目的
1	・UIパーツの同時操作 ・UIパーツの連打あるいは長押し ・UIパーツに意図しないデータを挿入 （指定とは違う文字あるいは文字列） ・会話やモーション動作中のUIパーツの操作	ロボアプリの場合、多くの不具合はディスプレイ側で起こることが多いため
2	・ユーザ視点 （例：ディスプレイは見やすいか（文字の大きさ、色のコントラスト、テキストや画像配置は適切か））	後続プロセスであるフィールドテストで不合格にならないようにするため （最終的な評価者であるお客様の確認は、単体テストでは得られないため、テストの合格基準が正しくない可能性はあるが、事前にテストを行うことで、誰が見ても明らかな問題をなくし手戻り回数を減らす）
3	（APIを使用している場合） ・使用APIによるBizPack提供サービスの動作・挙動への影響有無	API使用可否を判断するため （BizPack提供サービスの動作・挙動へ影響を与えていないことを確認する）

3.2.4.2. 単体テスト実施

ロボアプリ単体テスト仕様書に基づきテストを実施する。報告書では実施者、実施日時、結果を明記する。実施者、実施日を明記することにより、バグ修正時の問題解決に繋がるだけでなく、有効なエビデンスとなる。エビデンス取得が難しいテスト項目（例：モーション）は、ダブルチェックによりミスを防ぐ。

テストで不合格が出た場合は、テスト計画書記載の管理方法に従い対応する。

不具合の分析

不合格となったテスト項目を1件ごとに対応するだけでなく、不合格の発生件数について、全体の傾向を分析する。傾向分析の狙いは以下となる。

〈分析の狙い〉

- テスト項目に抜けや漏れがないか（テストは充分に行われているか）
- 特定分野に不備や仕様・設計ミスがないか

〈傾向分析例〉

		原因				
		仕様	定義／設定	入力処理	内部処理	出力処理
現象	a. 表示不正	1	1	3		
	b. 結果不正				14	5

	c. エラーチェック不正				6	
	d. 画面遷移不正				2	3
	e. モーション不正		5			
	f. 発話不正		2			
	g. 応答不正		1			7
	h. 異常終了					
	i. 操作不能					

不合格の偏りがあった場合に以下の仮説を確認する。

〈確認項目〉

- その分野の開発品質が悪いのではないか
- その分野の設計が曖昧ではないか
- 他の分野のテスト項目が不十分なのではないか（テスト不足により不合格が発生していない。設計漏れがあり、テスト項目が不足している等）

確認事項（単体テスト（ロボアプリ））		確認
1	テスト結果報告書にテスト実施日が記録されているか	
2	テスト結果報告書にテスト実施者が記録されているか	
3	テスト結果報告書にテスト結果が記録されているか	
4	エビデンス取得が難しい項目はダブルチェックが行われているか 例：モーション	
5	テスト計画書記載の終了条件を満たしているか	

3.2.4.3. 単体テスト結果レビュー・了承

テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。

3.2.4.4. 単体テスト結果報告書提出

開発会社内了承済みのテスト結果を、単体テスト仕様書兼結果報告書としてお客様へ提出する。

3.2.4.5. 単体テスト結果報告書受領・了承

開発会社から単体テスト仕様書兼結果報告書を受領・了承する。

3.3. 連携テスト

3.3.1. プロセス概要

連携テストプロセスでは、主に以下の作業を想定している。外部システム・外部機器（CMS・外部クラウド・その他）連携がある場合に実施する。

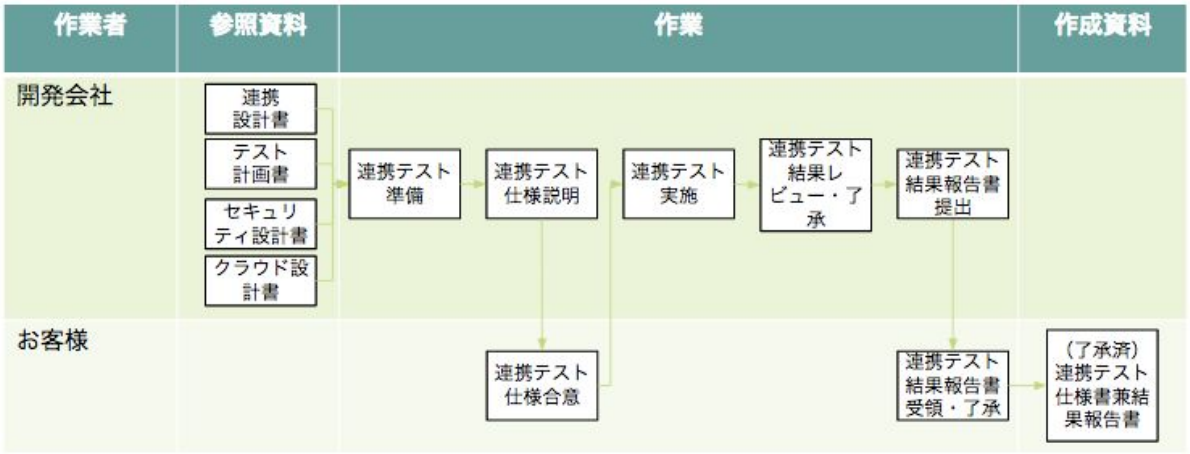
〈主な作業〉

- 連携設計書・テスト計画書・セキュリティ設計書・クラウド設計書を基に連携テスト仕様書を作成し、連携システムを担当するお客様にテスト仕様を説明、合意後にテストを実施する。
- テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。
- 開発会社内で了承済みのテスト結果を報告書としてお客様へ提出し、了承を得る。

3.3.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	連携設計書、テスト計画書、セキュリティ設計書、クラウド設計書
3	作成資料	連携テスト仕様書兼結果報告書
4	完了基準	お客様による連携テスト仕様書兼結果報告書の了承
5	後続プロセス	システムテスト

3.3.3. 作業の依存関係（連携テスト）



3.3.4. 作業詳細

3.3.4.1. 連携テスト準備

テスト計画書・セキュリティ設計書・クラウド設計書・連携設計書を基に、必要機材・ネットワーク環境を用意（ネットワーク疎通テスト含む）し、連携テスト仕様書を作成する。連携先が複数ある場合は、連携システム、連携機器ごとにテスト項目を作成する。
また異常系の処理では以下もテスト対象にする。

〈テスト対象例〉

- タイムアウト
- データフォーマットの誤り
- 入力データのバリデーションのチェック

テスト項目では、連携設計で整理したPepperとの連携エラーに対して、テスト項目を作成する。また、仕様以外のデータを入れてアプリが不具合を起こさないことを確認する。仕様書以外のデータは、入力されないと考えられるが、悪意のある第三者が意図的に不正なデータを入力することがあるため、必ずチェックを行う必要がある。

〈テスト項目例〉

- エラー対応：プリンタに出力依頼を出したが、応答がない。30秒後にPepperが「あれれ、ごめんなさい。プリンタの調子が悪いみたい。」と発話する。
- テスト項目：出力依頼30秒後に「あれれ、ごめんなさい。プリンタの調子が悪いみたい。」と発話しているか。

確認事項（連携テスト）		確認
1	連携先が複数ある場合、連携ごとにテスト項目を盛り込んでいるか	
2	Pepperとの連携エラーに対する対応をテスト項目に盛り込んでいるか	

3.3.4.2. 連携テスト仕様説明

テスト仕様書の説明をお客様に対して行う。

確認事項（連携テスト）		確認
1	対象とするテスト項目、システムをお客様に説明したか	
2	テスト期間中の対象システムに与える影響（負荷によるパフォーマンス低下・制限・障害等のリスク）をお客様に説明したか	
3	テスト期間中、対象システムを利用している担当者への業務影響をお客様に説明したか	
4	業務影響に対する対応方法をお客様に説明したか	
5	テスト実施期間・時間をお客様に説明したか	

6	テストデータの準備方法（誰が、どの時点のデータを準備するか、本番データ利用時のマスキング有無等）をお客様に説明したか	
7	テスト実施時にトラブルが発生した場合の対応方法をお客様に説明したか	
8	テスト終了後の回復方法（テストデータ消去、設定変更の戻し等）をお客様に説明したか	

3.3.4.3. 連携テスト仕様合意

連携テスト実施前にお客様から連携テスト実施の合意を得る。

3.3.4.4. 連携テスト実施

連携先システム側でトラブルが発生時の対応方法を実施前に確認し、連携テスト仕様書に基づきテストを実施する。報告書では実施者、実施日時、結果を明記する。

エビデンス取得が難しいテスト項目（例：モーション）は、ダブルチェックによりミスを防ぐ体制で実施する。連携システム側のテスト結果はお客様担当者へ取得を依頼する。

テストで不合格が出た場合は、テスト計画書記載の管理方法に従い対応する。

確認事項（連携テスト）		確認
1	テスト結果報告書にテスト実施日を記録しているか	
2	テスト結果報告書にテスト実施者を記録しているか	
3	テスト結果報告書にテスト結果を記録しているか	
4	エビデンス取得が難しい項目はダブルチェックを行っているか 例：モーション	
5	テスト計画書記載の終了条件を満たしているか	

3.3.4.5. 連携テスト結果レビュー・了承

テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。

3.3.4.6. 連携テスト結果報告書提出

連携テスト結果報告書をお客様へ提出する。

3.3.4.7. 連携テスト結果報告書受領・了承

開発会社から提出された連携テスト結果報告書をお客様が受領・了承する。

3.4. システムテスト

3.4.1. プロセス概要

システムテストプロセスでは、主に以下の作業を想定している。本テストでは本番環境と同等の環境を用意し動作を確認する。

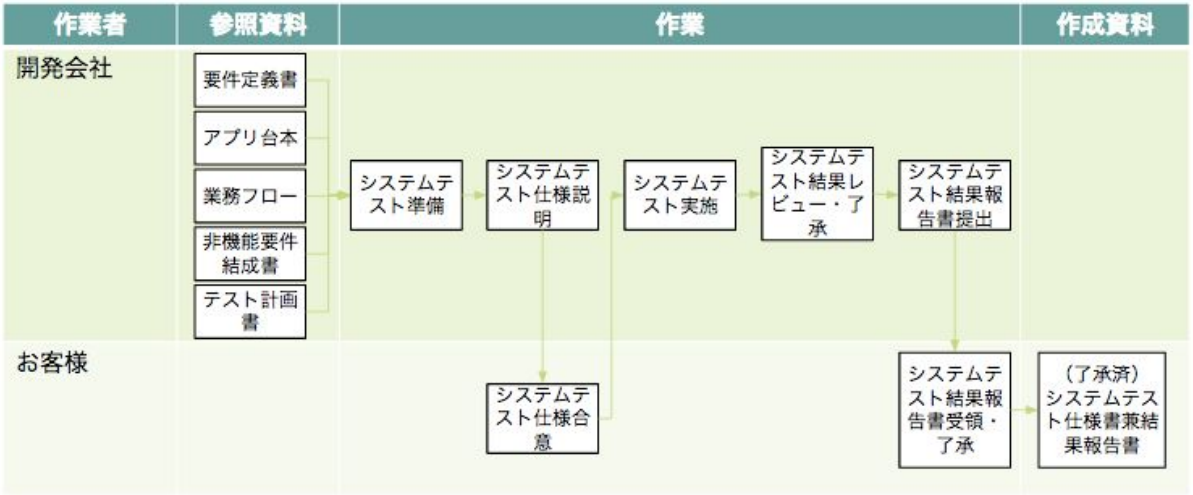
〈主な作業〉

- 要件定義書、アプリ台本、業務フロー、非機能要件設計書、テスト計画書を基にシステムテスト仕様書を作成し、お客様（運用管理者）へテスト仕様を説明、合意後にテストを実施する。
- テスト結果（エビデンス取得を含む）を開発会社の管理者がレビューする。
- 開発会社の社内レビュー済みのテスト結果を報告書としてお客様へ提出し、了承を得る。

3.4.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様（運用管理者）
2	参照資料	要件定義書、アプリ台本、業務フロー、非機能要件設計書、テスト計画書
3	作成資料	システムテスト仕様書兼結果報告書
4	完了基準	お客様によるシステムテスト仕様書兼結果報告書の了承
5	後続プロセス	フィールドテスト

3.4.3. 作業の依存関係（システムテスト）



3.4.4. 作業詳細

3.4.4.1.システムテスト準備

システムテストに向け、テスト計画書・要件定義書・アプリ台本・業務フロー・非機能要件設計書を基に開発会社がお客様と調整し必要な機材を用意し、システムテスト仕様書を作成する。結合を行う各システムの管理者にテスト実施の調整を行う。

テスト項目の作成では、業務フローの確認（正常フローのパターン・異常発生から復旧までのフロー）だけでなく、フィールドテストに向けてユーザ視点のテスト項目を入れ、問題がないか確認する。

ただし、ディスプレイは見やすいか等、本番環境に依存する項目は、フィールドテストで実施する。

また、要件定義書、非機能要件設計書よりセキュリティやパフォーマンスにかかわるテストは、本番環境に依存しない項目を作成する。

確認事項（システムテスト）		確認
1	業務フロー（正常フローのパターン・異常発生から復旧までのフロー）に関するテスト項目を盛り込んでいるか	
2	パフォーマンスに関するテスト項目を盛り込んでいるか 例： ①Pepperの応答速度（各種センサーの反応時間・連携機器の応答時間・連携システムの応答時間（WebAPI含む）） ②ロボアプリ処理性能（ビヘイビアの切替速度（ロード時間）・画面遷移の時間・画面UIパーツの反応速度（例：ボタンの反応速度は1秒を超えてないか）・内部処理ルーチンの回答速度（画像処理、暗号化処理等））	
3	ユーザ視点に関するテスト項目を盛り込んでいるか 例： ①ディスプレイ（押しやすいか・画面操作が多くないか） ②セリフ（質問は答えやすいか・応答速度は許容可能か・内容が理解しやすいか） ③モーション（人にストレスを与えたり危害を加えないか） ④シナリオ（短すぎたり長くないか・やりとりしていて疲れないか・余計なシナリオが入っていないか）	

3.4.4.2. システムテスト仕様説明

システムテスト仕様書の説明をお客様に対して行う。

確認事項（システムテスト）		確認
1	対象とするテスト項目、システムをお客様に説明したか	
2	テスト期間中の対象システムに与える影響（負荷によるパフォーマンス低下・制限・障害等のリスク）をお客様に説明したか	
3	テスト期間中、対象システムを利用している担当者への業務影響をお客様に説明したか	

4	業務影響に対する対応方法をお客様に説明したか	
5	テスト実施期間・時間をお客様に説明したか	
6	テストデータの準備方法（誰が、どの時点のデータを準備するか、本番データ利用時のマスキング有無等）をお客様に説明したか	
7	テスト実施時にトラブルが発生した場合の対応方法をお客様に説明したか	
8	テスト終了後の回復方法（テストデータ消去、設定変更の戻し等）をお客様に説明したか	

3.3.4.3. システムテスト仕様合意

システムテスト実施前にお客様とシステムテストの合意を得る。

3.4.4.4. システムテスト実施

トラブルが発生した場合の対応方法を実施前に確認し、システムテスト仕様書を基にシステムテストを実施する。報告書では実施者、実施日時、結果を明記する。

エビデンス取得が難しいテスト項目（例：モーション）は、ダブルチェックによりミスを防ぐ体制で実施する。連携システム側のテスト結果はお客様側担当者へ取得を依頼する。

テストで不合格が出た場合は、テスト計画書記載の管理方法に従い対応する。

確認事項（システムテスト）		確認
1	テスト結果報告書にテスト実施日を記録しているか	
2	テスト結果報告書にテスト実施者を記録しているか	
3	テスト結果報告書にテスト結果を記録しているか	
4	エビデンス取得が難しい項目はダブルチェックを行っているか 例：モーション	
5	テスト計画書記載の終了条件を満たしているか	

3.4.4.5. システムテスト結果レビュー・了承

テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。

3.4.4.6. システムテスト結果報告書提出

システムテスト仕様書兼結果報告書をお客様へ提出する。

3.4.4.7. システムテスト結果報告書レビュー・了承

開発会社から提出されたシステムテスト仕様書兼結果報告書をお客様が受領・了承する。

3.5. フィールドテスト

3.5.1. プロセス概要

フィールドテストプロセスでは、主に以下の作業を想定している。本テストでは関連する本番システムを結合させ動作を確認する。

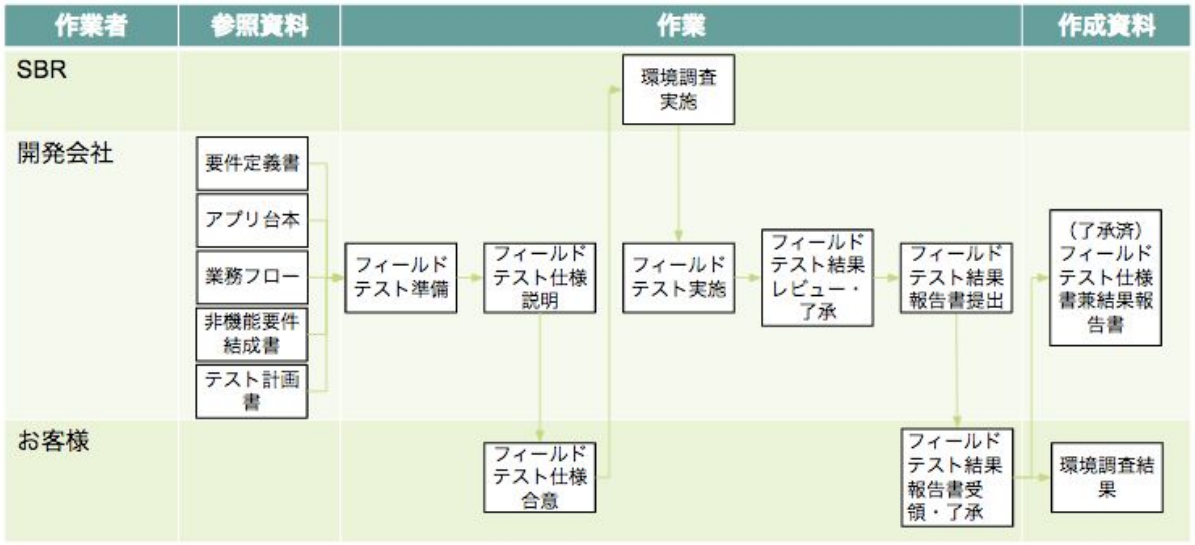
〈主な作業〉

- 要件定義書、アプリ台本、業務フロー、非機能要件設計書、テスト計画書を基にフィールドテスト仕様書を作成し、お客様（運用管理者及び現場オペレータ）へテスト仕様を説明、合意後にテストを実施する。
- フィールドテストに先立ち、環境調査（詳細な実地調査）を実施し、設置場所候補がチェックリストの要綱を全て満たしていることを確認する。
- テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。
- 開発会社で社内レビュー済みのテスト結果を報告書としてお客様へ提出し、了承を得る。

3.5.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様（運用管理者、現場オペレータ）
2	参照資料	要件定義書、アプリ台本、業務フロー、非機能要件設計書、テスト計画書
3	作成資料	フィールドテスト仕様書、結果報告書、環境調査結果
4	完了基準	お客様によるフィールドテスト仕様書兼結果報告書の了承
5	後続プロセス	UAT

3.5.3.作業の依存関係（フィールドテスト）



3.5.4. 作業詳細

3.5.4.1. フィールドテスト準備

テスト計画書を基に、フィールドテストに向け必要リソース（開発会社・お客様双方のテスト実施者、テスト実施場所、機材及びシステム）、テスト仕様書を用意する。テスト項目では、業務フローの確認（正常フローのパターン網羅、システム異常発生から復旧まで等、異常フローの網羅）だけでなく、ユーザ視点のテスト項目を入れ、問題ないか確認する。

また以下を考慮し、システムテストで確認が難しい本番環境に関するテスト項目も確認する。

〈考慮点〉

- エンドユーザの変化（日中は子供、夜間は社会人等、エンドユーザ属性が変わる）
- 環境の変化（時間帯・平日・休日・繁忙期・閑散期での人の多さ・騒音・光量等）

〈確認例〉

- ディスプレイは見やすいか（色のコントラストが照明の影響で見えにくくなっていないか）
- 撮影した画像に問題がないか（顔認証に利用可能か）
- Pepperが照明の方向を向いていないか
- 通信に支障がないか
- （運用として）充電等を実施可能か
- （お客様がPepperを既に導入済の場合）他のロボアプリへの影響はないか、また影響を受けていないか

異常フローのテストでは、本番環境では異常発生させられないケースが多いため、テスト項目を用意しても実施できない場合がある。その場合、異常が発生した想定で、復旧までをテストする等、代替テストシナリオの用意とお客様との合意が必要となる。

確認事項（フィールドテスト）		確認
1	フィールドテストの確認ポイント、テストシナリオを定義しているか	
2	フィールドテストの実施リソース（開発会社・お客様双方のテスト実施者、テスト実施場所、機材及びシステム）を確保しているか	
3	業務フロー（正常フローのパターン・異常発生から復旧までのフロー）に関するテスト項目を盛り込んでいるか	
4	ユーザ視点に関するテスト項目を盛り込んでいるか 例： ①ディスプレイ（見やすいか・押しやすいか・画面操作が多くないか） ②セリフ（聞き取りやすいか・質問は答えやすいか・応答速度は許容可能か・内容が理解しやすいか） ③モーション（人にストレスを与えたり危害を加えないか） ④シナリオ（短すぎたり長くないか・やりとりしていて疲れないか・余計なシナリオが入っていないか・使い方がわからない利用者が利用して安全か）	
5	異常発生させられない場合、代替テストシナリオ（異常が発生した想定で復旧までをテストする等）を用意しているか	

3.5.4.2. フィールドテスト仕様説明

フィールドテスト仕様書の説明をお客様に対して行う。

確認事項（フィールドテスト）		確認
1	対象とするテスト項目、システムをお客様に説明したか	
2	テスト期間中の対象システムに与える影響（負荷によるパフォーマンス低下・制限・障害等のリスク）をお客様に説明したか	
3	テスト期間中、対象システムを利用している担当者への業務影響をお客様に説明したか	
4	業務影響に対する対応方法をお客様に説明したか	
5	テスト実施期間・時間をお客様に説明したか	
6	テストデータの準備方法（誰が、どの時点のデータを準備するか、本番データ利用時のマスキング有無等）をお客様に説明したか	
7	テスト実施時にトラブルが発生した場合の対応方法をお客様に説明したか	
8	テスト終了後の回復方法（テストデータ消去、設定変更の戻し等）をお客様に説明したか	

3.5.4.3. フィールドテスト仕様合意

トラブルが発生した場合の対応方法を実施前に確認し、フィールドテストを実施する。

3.5.4.4. 環境調査実施

設置候補場所の環境調査を実地でSBRが行う。

3.5.4.5. フィールドテスト実施

トラブル発生時の対応方法を実施前に確認し、フィールドテスト仕様書に基づきフィールドテストを実施する。報告書では実施者、実施日時、結果を明記する。

エビデンス取得が難しいテスト項目（例：モーション）は、ダブルチェックによりミスを防ぐ体制で実施する。連携システム側のテスト結果はお客様側担当者へ取得を依頼する。

テストで不合格が出た場合は、テスト計画書記載の管理方法に従い対応する。

モンキーテスト

モンキーテストとは思いついたことを手当たり次第行うテストのことである。場当たり的に操作を行う。

フィールドテストは開発者でない利用者が操作することが多い。システムのことを全くしらない利用者にシステムを使ってもらうことで、通常テストでは検出できないシステムの不具合をあぶり出せる。

不具合が発見した場合は、開発側にフィードバックが必要となる。開発者が不具合対応をするために、不具合発生手順を具体的に明文化することが望ましいが、モンキーテストでは、場当たり的に操作するため、不具合発生手順を再現できないこともある。その場合、ログ情報から不具合発生手順を再現できることがあるため、不具合発生時刻を記録しておく。

開発へのフィードバッグ

開発したアプリを実環境でテストすることで様々な問題が浮かび上がってくる。そのため、実用的な品質になるまでテスト結果を開発へフィードバックする。

確認事項（フィールドテスト）		確認
1	テスト結果報告書にテスト実施日を記録しているか	
2	テスト結果報告書にテスト実施者を記録しているか	
3	テスト結果報告書にテスト結果を記録しているか	
4	エビデンス取得が難しい項目はダブルチェックを行っているか 例：モーション	
5	フィードバック結果を開発要件として落とし込めているか	
6	テスト計画書記載の終了条件を満たしているか	

3.5.4.6. フィールドテスト結果レビュー・了承

テスト結果（エビデンス取得を含む）を開発会社の管理者がレビュー・了承する。

3.5.4.7. フィールドテスト結果報告書提出

フィールドテスト仕様書兼結果報告書をお客様へ提出する。

3.5.4.8. フィールドテスト結果報告書レビュー・了承

開発会社から提出されたフィールドテスト仕様書兼結果報告書をお客様が受領・了承する。

4. リリース

リリースフェーズは、開発・テストが完了し、運用へ向けて準備作業を行うフェーズである。同フェーズは審査、教育、メンテナンス準備という3つのカテゴリーに分かれる。審査では、UATを経て安全性審査、脆弱性診断を実施する。教育では、お客様側でPepper運用管理者へのトレーニング、Pepperを扱う現場オペレータへのユーザトレーニングの2本立てで行う。またメンテナンス準備では、開発会社とお客様の間でSLA（サービスレベルアグリーメント）を含む運用に関する契約と契約に基づく環境準備を行う。最後に実環境での運用開始可否を判定する。

本フェーズは「4.1 UAT」「4.2 脆弱性診断」「4.3 安全性審査」「4.4 管理者トレーニング」「4.5 ユーザトレーニング」「4.6 メンテナンス準備」「4.7 リリース判定」の7プロセスとなっている。

4.1. UAT

4.1.1. プロセス概要

UAT（User Acceptance Test）プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 導入提案書、要件定義書、テスト計画書を基にお客様にてUAT仕様書を作成し、開発会社と合意後にテストを実施する。
- テスト結果（エビデンス取得を含む）をお客様がレビューし、承認する。

4.1.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様（Pepper運用管理者・現場オペレータ）
2	参照資料	導入提案書、要件定義書、テスト計画書
3	作成資料	UAT結果
4	完了基準	お客様によるUAT結果の承認
5	後続プロセス	脆弱性診断

4.1.3. 作業の依存関係（UAT）

4.1.4. 作業詳細

4.1.4.1. UAT準備

お客様が発注したシステムが導入提案書・要件定義書・テスト計画書に従い期待通りに動くかを確認する。テスト項目はお客様自身が作成する。

〈確認観点例〉

- フィールドテストとの比較では、より提案書記載のKGI・KPI達成の妥当性判断に重きが置かれる
    - ロボアプリ（システム）が正常に動作するか
    - ロボアプリ（システム）が業務要件に適合しているか
    - 業務遂行上で問題はないか

確認事項（UAT）		確認
1	納品時、お客様に起動するビヘイビアパスを通知したか （ビヘイビアパスは<appid>/<behaviorのある場所>で記載 （例1）appidがpepper_appで、pepper_app直下にbehavior.xarがある場合、ビヘイビアパスは、pepper_app/.となる （例2）appidがpepper_appで、pepper_app/behavior_1以下にbehavior.xarがある場合、ビヘイビアパスは、pepper_app/behavior_1となる ※ビヘイビアパスは、「お仕事かんたん生成」で登録する際に必要）	

4.1.4.2. UAT仕様確認・合意

お客様が作成したUATテスト仕様（項目）を確認し、UAT実施にあたり開発会社側で関与する項目があるかを確認する。

〈確認項目例〉

- UAT環境の整備
- UAT実施担当者への事前トレーニング
- 実施中のトラブル対応

確認事項（UAT）		確認
1	お客様が作成したUATテスト仕様（項目）からUAT環境の整備及び実施中のトラブル対応等、開発会社側で関与すべき項目があるか確認したか	

4.1.4.3. UAT実施

お客様がUATを実施する。

4.1.4.4. UAT結果レビュー・承認

お客様側でUATの結果をレビュー・承認する。

4.1.4.5. UAT結果フィードバック

お客様がUAT結果を開発会社にフィードバックし、両社で対応方法、スケジュールの確認を行い合意・対応する。

確認事項（UAT）		確認
1	お客様からのフィードバックを受け、対応の実施可否（瑕疵に該当しないものの対応を決める）について合意を得ているか	
2	対応スケジュールについて合意を得ているか	
3	対応スケジュールに基づき、対応を完了させたか	

4.2. 脆弱性診断

4.2.1. プロセス概要

脆弱性診断プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 第三者機関によってWeb連携システム（CMS、WebAPIを通じて接続する外部システム等）のWeb脆弱性診断を実施し、結果が全て「PASS」であることを確認する。

4.2.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、第三者機関
2	参照資料	UAT結果
3	作成資料	脆弱性診断結果
4	完了基準	脆弱性診断の結果が全て「PASS」であること
5	後続プロセス	安全性審査

4.2.3. 作業の依存関係（脆弱性診断）

4.2.4. 作業詳細

4.2.4.1. 脆弱性診断依頼

第三者機関にWeb連携システムのWeb脆弱性診断の依頼を行う。依頼にあたっては診断項目としてネットワーク脆弱性診断とWebアプリケーション脆弱性診断を実施する。

Pepperで取得した個人情報等をCMSに格納する場合は必ず実施する。

〈ネットワーク脆弱性診断項目例〉

診断項目	
1	ホストのスキャン
2	ネットワークサービスの脆弱性
3	Webサーバの脆弱性
4	各種OSの脆弱性
5	悪意あるソフトウェア
6	ネットワーク機器の脆弱性
7	その他（その他ホスト全体の調査）

〈Webアプリケーション脆弱性診断項目例〉

診断項目	
1	認証
2	セッション管理
3	入出力処理
4	一般的な脆弱性
5	Webサーバ設定
6	認可

診断実施日は、サーバへの悪意ある攻撃や異常検知と認識しないようサーバ運用担当者・企業と調整を行い決定する。また、業務影響が発生しないようお客様関係部署と調整を行い決定する。

4.2.4.2. 脆弱性診断実施

第三者機関が脆弱性診断を実施し、申請者に結果を報告する。

4.2.4.3. 脆弱性診断結果受領

第三者機関から脆弱性診断の結果を受領する。
結果をお客様と共有し、問題がある場合、修正対象・修正内容・修正スケジュールを確認し、お客様との合意を受け、問題箇所を修正し、再度脆弱性診断を行う。

確認事項（脆弱性診断）		確認
1	脆弱性診断に合格しているか	

4.3. 安全性審査

4.3.1. プロセス概要

安全性審査プロセスでは、主に以下の作業を想定している。

〈主な作業〉

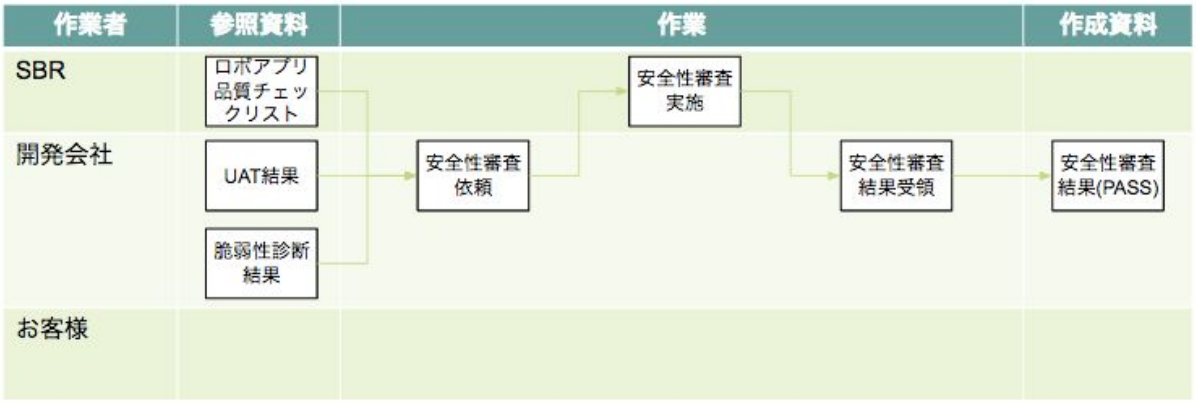
- ロボアプリ品質チェックリスト記載のテストを行っていることを確認し、結果が全て問題がないことを確認する。
- SBR担当部門（Pepper パートナー会員事務局）によるロボアプリ安全性審査[10]を実施し、結果が全て「PASS」であることを確認する。

4.3.2. 作業者と入出力確認表

項目		説明
1	作業者	SBR（Pepper パートナー会員事務局）、開発会社
2	参照資料	ロボアプリ品質チェックリスト、UAT結果、脆弱性診断結果
3	作成資料	ロボアプリ安全性審査結果
4	完了基準	ロボアプリ安全性審査の結果が全て「PASS」であること
5	後続プロセス	管理者トレーニング、ユーザトレーニング、メンテナンス準備

[10] http://www.softbank.jp/robot/developer/dev-support/program/partner/benefit/#03 参照

4.3.3. 作業の依存関係（安全性審査）



4.3.4. 作業詳細

4.3.4.1. 安全性審査依頼

SBRのPepper パートナープログラム事務局を通じて、ロボアプリ安全性審査を依頼する。依頼時には、改めて以下の確認事項（安全性審査）の基準を満たしているか確認し、1つでも要求を満たしていない場合は、お客様から合意を得た上で、要求を満たすようにアプリを修正する。

本申請はPepperパートナープログラム[11]（以下PPP）認定企業のみが行える。申請方法は、認定取得とともに入手できるPepperパートナープログラム認定企業向け『パートナー向けマニュアル』を参照。

安全性審査を受けるメリットと注意点は以下となる。

〈メリット〉

- 審査済みロボアプリが故障の起因となった場合、お客様が負担する修理費用が免除される
- 安全性審査を受けてないロボアプリとの差別化が図れる

〈注意点〉

- ロボアプリの審査や改修時の再審査には審査費用が発生する
- 開発・テストフェーズ終了後の申請では、他の開発会社の審査件数によっては審査待ちになる可能性があるため、要件定義終了後に申請する等、余裕を持って申請する（ただし審査完了希望日までの審査完了を保証するものではない）。

確認事項（安全性審査）		確認
1	ロボアプリ品質チェックリストの項目の要求を全て満たしているか	

[11] http://www.softbank.jp/robot/developer/dev-support/program/partner/

4.3.4.2. 安全性審査実施

SBRがロボアプリ安全性審査を実施し、申請者に結果を報告する。

4.3.4.3. 安全性審査結果受領

SBRからロボアプリ安全性審査の結果を受領する。
結果をお客様と共有し、問題がある場合、修正対象・修正内容・修正スケジュール・再テスト（連携テスト、システムテスト、フィールドテスト）の実施可否を確認し、お客様との合意を受け、問題箇所を修正し、再度ロボアプリ安全性審査申請を行う。

確認事項（安全性審査）		確認
1	ロボアプリ安全性審査を合格しているか	

4.4. 管理者トレーニング

4.4.1. プロセス概要

管理者はお客様側のPepper運用管理者を指す。管理者トレーニングプロセスでは、主に以下の作業を想定している。

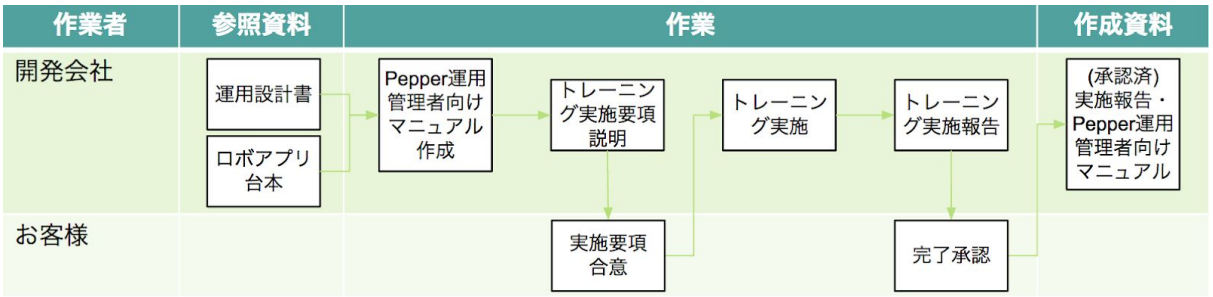
〈主な作業〉

- シナリオ設計書、運用設計書を基に、Pepper運用管理者向けマニュアルを作成し、トレーニング実施要項（トレーニング内容、スケジュール、実施方法等）を作成し、お客様へ説明し合意を得た上で、Pepper運用管理者向け管理者トレーニングを実施する。
- 実施後、お客様から完了の承認を得る。

4.4.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様（Pepper運用管理者、連携システム運用管理者）
2	参照資料	運用設計書、シナリオ設計書
3	作成資料	管理者トレーニング実施報告書・Pepper運用管理者向けマニュアル
4	完了基準	お客様による管理者トレーニング実施報告書の承認
5	後続プロセス	リリース判定

4.4.3. 作業の依存関係（管理者トレーニング）



4.4.4. 作業詳細

4.4.4.1. Pepper運用管理者向けマニュアル作成

Pepper運用管理者向けマニュアルを作成する。マニュアルの主な内容は以下となる。

お客様が初めてPepperを利用する場合は、Pepperの取り扱いや安全上、使用上の注意点が記載されたマニュアルを別途用意する。

〈マニュアル記載内容例〉

必須・任意		記載内容		備考
1	必須	a	運用フロー（通常業務）	通常業務及びトラブル時の対応
		b	エラーコード対応表兼トラブルシューティング集	
		c	トラブル発生時の問い合わせ	・お客様から開発会社への問い合わせ先 ・問い合わせ対応時間 ・連絡方法
2	任意	d	マニュアル情報	参照： http://www.softbank.jp/robot/biz/support/document/
		e	安全上のご注意	参照： http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-01.html
		f	使用上のご注意	参照： http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-02.html

現場オペレータからPepper運用管理者へ問い合わせが来た際に、Pepper運用管理者が問い合わせ内容を理解できるようにするため、ユーザマニュアル（後続プロセスでユーザトレーニングとして使用する現場オペレータ向けマニュアル）の準備も併せて行う。

確認事項（管理者トレーニング）		確認
1	Pepper運用管理者向けマニュアルには通常業務・トラブル発生時の対応フローを記載しているか	
2	Pepper運用管理者向けマニュアルにはエラーコード対応表兼トラブルシューティング集、トラブル発生時の問い合わせ先を記載しているか	
3	現場オペレータ側の操作も把握できるようユーザマニュアルの準備をしているか	

4.4.4.2. トレーニング実施要項説明

トレーニングの実施要項（トレーニング対象、内容、スケジュール、実施方法等）を検討する。トレーニングアジェンダは以下の点を考慮し決定する。

〈考慮点〉

- Pepperに初めて触るか
    - 初めての場合は、Pepperの取り扱い、お仕事かんたん生成の使い方（ネットワーク接続・アップデート・お仕事選択・お仕事登録方法）等の開発対象外のトレーニングを求められる可能性が高い
- Pepper運用管理者は現場オペレータからの問い合わせ一次受付先となるか

〈トレーニングアジェンダ例〉

- 本プロジェクトについて、Pepper導入の目的
- システム概要
- Pepperの取り扱い
- お仕事かんたん生成の使い方
- マイアプリ配信管理
- マイアプリ使用方法
- インタラクション分析の見方
- トラブルシューティング
- 故障交換時の対応
- 安全上、使用上の注意点

お仕事かんたん生成のトレーニング等、マイアプリ以外のトレーニングを実施する場合は、バージョンアップやリニューアルが実施されていないか確認し、最新情報を内容に反映させる。

〈リリース情報〉

- http://www.softbank.jp/robot/biz/cloud-service/#news
- http://www.softbank.jp/robot/biz/releasenotes/

また、連携システム側で新たな運用が発生する場合は、連携システムの運用管理者に対してトレーニングを実施するか否か検討する。

お客様及びPepper運用管理者に実施要項を説明する。

確認事項（管理者トレーニング）	確

		認
1	トレーニングアジェンダはトレーニング対象者のスキルレベル（Pepper運用経験の有無）と合致しているか？	

4.4.4.3. 実施要項合意

トレーニング実施についてお客様と内容に問題がないか確認し、合意する。

確認事項（管理者トレーニング）		確認
1	実施要項（トレーニング内容、スケジュール、実施方法）を説明し、理解と合意を得られたか	

4.4.4.4. トレーニング実施

Pepper運用管理者向けのトレーニングを実施する。また現場オペレータからの問い合わせに対応するために、別途ユーザトレーニングで使用するユーザマニュアルも提供する。

確認事項（管理者トレーニング）		確認
1	トレーニングを実施し、受講者の理解を得られたか	

4.4.4.5. トレーニング実施報告

トレーニング実施後、トレーニング実施要項の内容を説明した実施報告書をお客様へ提出する。

4.4.4.6. 完了承認

開発会社からトレーニング実施報告書を受領・完了承認する。

4.5. ユーザトレーニング

4.5.1. プロセス概要

ユーザトレーニングプロセスでは、主に以下の作業を想定している。

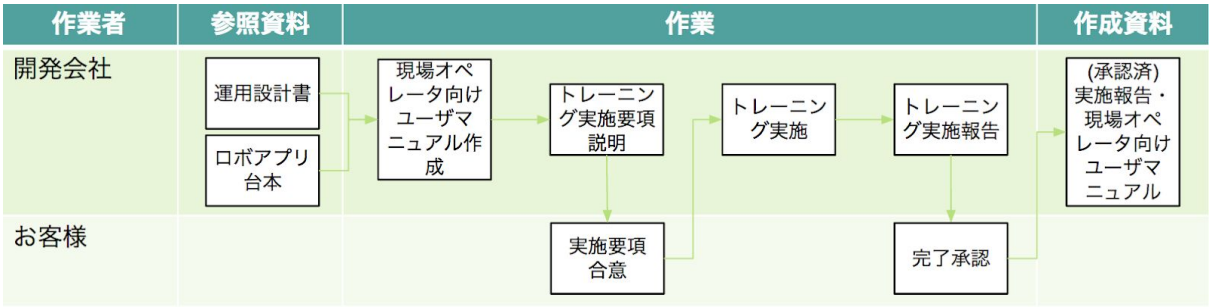
〈主な作業〉

- シナリオ設計書、運用設計書を基に、現場オペレータ向けのユーザマニュアルを作成し、トレーニング実施要項（トレーニング内容、スケジュール、実施方法等）をお客様へ説明し合意を得た上で、現場オペレータ向けユーザトレーニングを実施する。
- 実施後、お客様から完了の承認を得る。

4.5.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様（Pepper運用管理者、現場オペレータ）
2	参照資料	運用設計書、シナリオ設計書
3	作成資料	ユーザトレーニング実施報告書・現場オペレータ向けユーザマニュアル
4	完了基準	お客様によるユーザトレーニング実施報告書の承認
5	後続プロセス	リリース判定

4.5.3. 作業の依存関係（ユーザトレーニング）



4.5.4. 作業詳細

4.5.4.1. 現場オペレータ向けユーザマニュアル作成

現場オペレータ向けのユーザマニュアルを作成する。通常運用だけでなく、トラブル発生時の対応フローをユーザマニュアルに記載する。また、外部システム、外部機器等連携システムがある場合、連携システムについてもユーザマニュアルに記載する。

〈マニュアル記載内容例〉

必須・任意		記載内容		備考
1	必須	a	運用フロー	・通常業務及びトラブル時の対応、さらに、連携システムも対象に含める。
		b	エラーコード対応表兼トラブルシューティング集	
		c	トラブル発生時の問い合わせ	・お客様からPepper運用管理者への問い合わせ先 ・問い合わせ対応時間 ・連絡方法
2	任意	d	マニュアル情報	参照：

				http://www.softbank.jp/robot/biz/support/document/
		e	安全上のご注意	参照： http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-01.html
		f	使用上のご注意	参照： http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-02.html

確認事項（ユーザトレーニング）		確認
1	現場オペレータ向けのユーザマニュアルには通常業務・トラブル発生時の対応フローを記載しているか	
2	現場オペレータ向けのユーザマニュアルにはエラーコード対応表兼トラブルシューティング集、トラブル発生時の問い合わせ先を記載しているか	

4.5.4.2. トレーニング実施要項説明

お客様にトレーニングの実施要項（トレーニング内容、スケジュール、実施方法等）を説明する。

〈トレーニングアジェンダ例〉

- 本プロジェクトについて、Pepper導入の目的
- Pepperの開梱・梱包
- Pepperの基本操作（電源ON/OFF、移動、充電の仕方について等）
- Pepperの各アプリ概要
- お仕事かんたん生成の使い方（ネットワーク接続・アップデート・お仕事選択）
- マイアプリ利用方法
- トラブルシューティング
- 故障交換時の対応
- 安全上、使用上の注意点

確認事項（ユーザトレーニング）		確認
1	トレーニングアジェンダはトレーニング対象者のスキルレベル（Pepper運用経験の有無）と合致しているか。	

4.5.4.3. 実施要項合意

トレーニングの実施についてお客様と内容に問題がないか確認し、合意する。

確認事項（ユーザトレーニング）	確

		認
1	実施要項（トレーニング内容、スケジュール、実施方法）を説明し、理解と合意を得られたか	

4.5.4.4. トレーニング実施

現場オペレータ向けのトレーニングを実施する。

確認事項（ユーザトレーニング）		確認
1	トレーニングを実施し、受講者の理解を得られたか	

4.5.4.5. トレーニング実施報告

トレーニング実施後、実施要項の内容を説明したトレーニング実施報告書をお客様へ提出する。

4.5.4.6. 完了承認

開発会社からトレーニング実施報告書を受領・完了承認する。

4.6. メンテナンス準備

4.6.1. プロセス概要

メンテナンス準備プロセスでは、主に以下の作業を想定している。

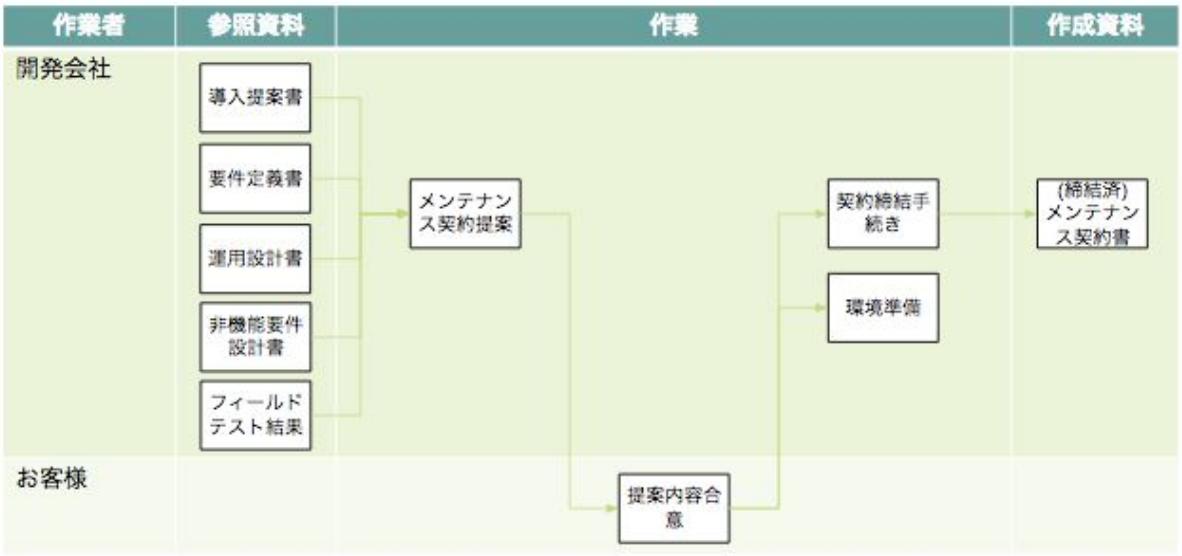
〈主な作業〉

- メンテナンス契約の提案をお客様へ提示し、お客様合意後、契約手続きを行う。
- メンテナンス契約に基づき、メンテナンスで必要な環境の準備を行う。

4.6.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	導入提案書、要件定義書、運用設計書、非機能要件設計書、フィールドテスト結果報告書
3	作成資料	メンテナンス契約書
4	完了基準	お客様とのメンテナンス契約締結
5	後続プロセス	リリース判定

4.6.3. 作業の依存関係（メンテナンス準備）



4.6.4. 作業詳細

4.6.4.1. メンテナンス契約提案

メンテナンス契約書を用意し、お客様へ提案する。メンテナンス契約は以下のようなサービスが想定される。

〈サービス例〉

- 専用ヘルプデスク
- レポート提供とその内容（メンテナンス報告書等）
- ソフトウェア保守（不具合修正、バージョンアップ、脆弱性診断の継続実施等）
- サーバ監視
- 障害対応
- 障害時のオンサイト対応（障害ネットワーク機器の交換等）

上記はサービスレベル（定義、範囲、内容、達成目標等）を明確化する必要がある。
またPepperのサポート範囲・内容に対して、開発したロボアプリのサポート範囲・内容が重複や抜け漏れがないようにする。

Pepperサポート：http://www.softbank.jp/robot/biz/support/

今後、セリフ等の修正が多発する、定期的にロボアプリ保守が発生する等が見込まれる場合は、その対応（サービスメニュー表及びメニュー単価を決め発生件数で月次請求、都度見積にする等）を契約に盛り込み提案する。

確認事項（メンテナンス準備）	確認

1	サービス内容とサービスレベルを明記しているか	
2	Pepperのサポート範囲・内容とロボアプリのサポート範囲・内容に重複や抜け漏れがないか	

4.6.4.2. 提案内容合意

提案した契約内容をお客様と確認し、合意する

4.6.4.3. 契約締結手続き

契約締結に向け、手続きを進める。

確認事項（メンテナンス準備）		確認
1	開発会社側で、法務チェックを受けているか	

4.6.4.4. 環境準備

メンテナンス契約書を基に、メンテナンスに必要な環境の準備を行う。

〈準備例〉

- 専用ヘルプデスク開設（要員確保、回線用意等）
- オンサイト対応（要員確保等）
- サーバ監視（体制構築等）
- 障害対応（体制構築等）

環境準備と共に運用設計で作成した運用フローの更新を行う。
開発会社が専用ヘルプデスクを開設する場合、トラブル発生時に問題の切り分けによって問い合わせ先が異なる形になるが、問題の切り分けを行うことができない場合は、以下の問い合わせ順にすることでPepper運用管理者の問い合わせ負担を軽減する。

〈問い合わせ順〉

1. 現場オペレータからPepper運用管理者へ問い合わせ
2. Pepper運用管理者から開発会社の専用ヘルプデスクへ問い合わせ
3. Pepper運用管理者からPepperのサポートへ問い合わせ

トラブル発生時、現場オペレータ、Pepper運用管理者が問題の切り分けを行うことができず問い合わせを行う場合、アプリを起因とした不具合は、Pepperのサポートでは対応できない。結果として、詳細なやりとりを行ったにもかかわらず、対応できないと回答する形となり、Pepper運用管理者は、再度開発会社へ問い合わせることになり問い合わせ工数が増加する。
従って、最初に開発会社へ問い合わせを行い、開発会社がアプリ起因か機体起因か切り分けを行う形にして、Pepper運用管理者の問い合わせ負担を軽減する。

4.7. リリース判定

4.7.1. プロセス概要

リリース判定プロセスでは、主に以下の作業を想定している。

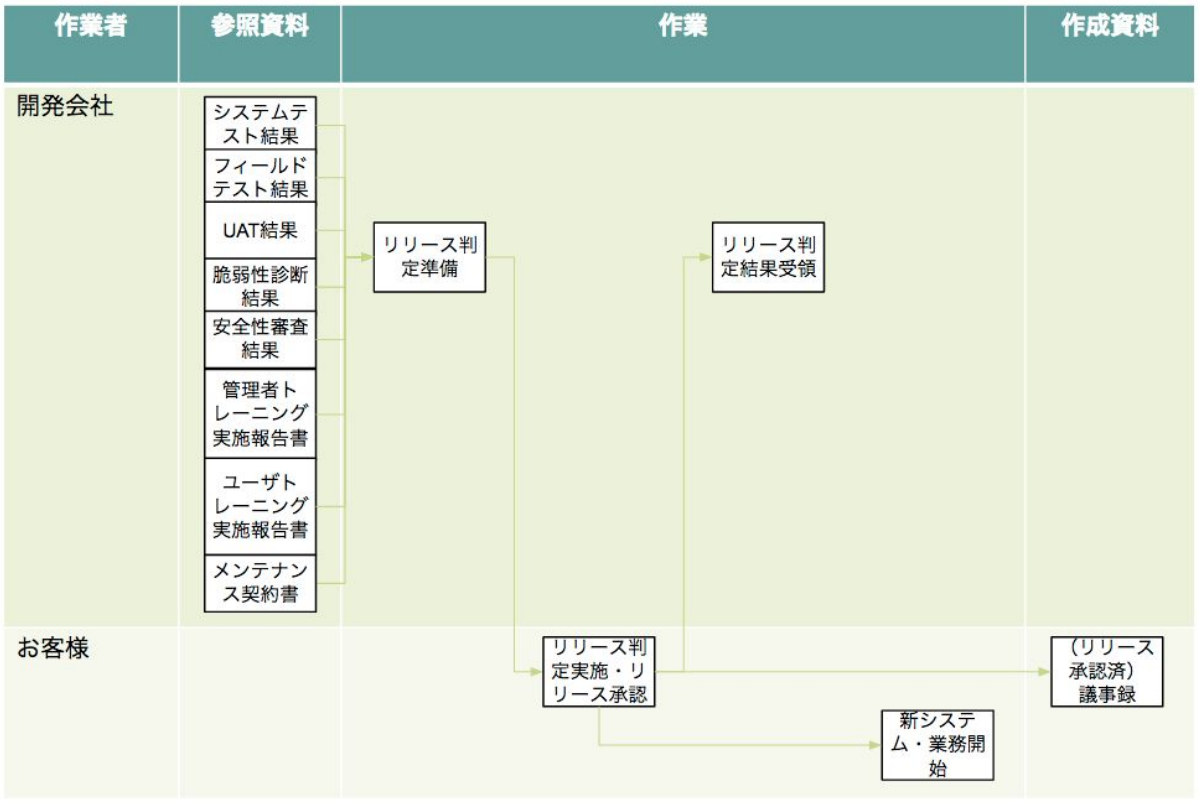
〈主な作業〉

- 事前に用意されたリリース判定基準を基にお客様側でリリース判定会議を開催し、判定を行う。
- リリース判定に合格した場合は新システム・業務を開始する。

4.7.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社、お客様
2	参照資料	システムテスト結果報告書、フィールドテスト結果報告書、UAT結果、脆弱性診断結果、安全性審査結果、管理者トレーニング実施報告書、ユーザトレーニング実施報告書、メンテナンス契約書
3	作成資料	議事録（リリース承認済）
4	完了基準	お客様によるリリースの承認
5	後続プロセス	効果測定

4.7.3. 作業の依存関係（リリース判定）



4.7.4. 作業詳細

4.7.4.1. リリース判定準備

リリース判定に必要な準備物を用意する。準備に必要なものはリリース判定基準に則って用意する。主な判定基準、準備物は以下となる。

〈リリース判定基準例〉

- 開発品質
    - 各テスト・審査・診断結果
    - 課題管理表、バグ管理表等でリリースにあたって問題となる課題、バグが残っていないかお客様と確認し対応する
- 新業務フローの整備完了状況
- Pepper運用管理者・現場オペレータの習熟度
- 社内外の調整状況、アナウンス状況
- 運用保守体制
    - リリース初期の初動対応、体制含む
- コンティンジェンシープラン（リリース開始できなかった場合のリカバリー対応）

〈準備物例〉

- システムテスト結果報告書
- フィールドテスト結果報告書

- UAT結果
- 脆弱性診断結果
- 安全性審査結果
- メンテナンス契約書
- 管理者トレーニング実施報告書
- ユーザトレーニング実施報告書
- 運用保守体制

リリース判定後、速やかな導入に移すために必要であれば、Pepper運用管理者向けマニュアル、現場オペレータ向けユーザマニュアルを抜粋した導入手順書を用意する。

4.7.4.2. リリース判定実施・リリース承認

準備した審査・診断結果及び契約内容を確認し、リリース判定を行う。問題がなければ、お客様がリリース承認を行う。

確認事項（リリース判定）		確認
1	お客様からリリース承認を受けたか	

4.7.4.3. リリース判定結果受領

リリース判定結果をお客様から受領する。

4.7.4.4. 新システム・業務開始

お客様からのリリース承認を基に、新システム稼動を行い、新業務を開始する。

5. 運用

運用フェーズは、各テストを通過したロボアプリ（新システム・業務含む）を実環境でリリースし、運用開始するフェーズである。

運用はメンテナンス契約書に従う。導入提案書で定めたタイミングで効果測定を行い、効果測定結果を基にお客様にシナリオ修正を提案し、運用を続けていく。

本フェーズは「5.1 効果測定」「5.2 メンテナンス」の2プロセスとなっている。

5.1. 効果測定

5.1.1. プロセス概要

効果測定プロセスでは、主に以下の作業を想定している。

〈主な作業〉

- 目標設定に基づいた導入効果測定を実施し、お客様に結果を報告する。

5.1.2. 作業者と入出力確認表

項目		説明
1	作業者	開発会社
2	参照資料	導入提案書
3	作成資料	導入効果報告書
4	完了基準	お客様による導入効果報告書の承認
5	後続プロセス	メンテナンス

5.1.3. 作業の依存関係（効果測定）

5.1.4. 作業詳細

5.1.4.1. 導入効果測定

企画プロセスの目標設定で定めたタイミングで導入効果測定を行う。測定結果を導入効果報告書にまとめる。

効果測定の際に、併せてお客様及びエンドユーザからアンケートを取る。アンケート実施のメリットは以下となる。

〈メリット〉
- ポジティブな声
    - 効果測定だけでなく、副次的な効果が把握できる。
    - より効果的な使い方を検討する参考材料となる。
- ネガティブな声
    - 今後の改善ポイントが把握できる。

また、アンケート集計した声をチャート化して、全体の傾向も把握する。

確認事項（効果測定）		確認
1	導入効果測定を行ったか	

5.1.4.2. 導入効果報告書提出

作成した導入効果報告書を、お客様に提出する。

5.1.4.3. 導入効果報告書受領・承認

開発会社から提出された導入効果報告書を受領・承認する。

5.2. メンテナンス

5.2.1. プロセス概要

メンテナンスプロセスでは、主に以下の作業を想定している。

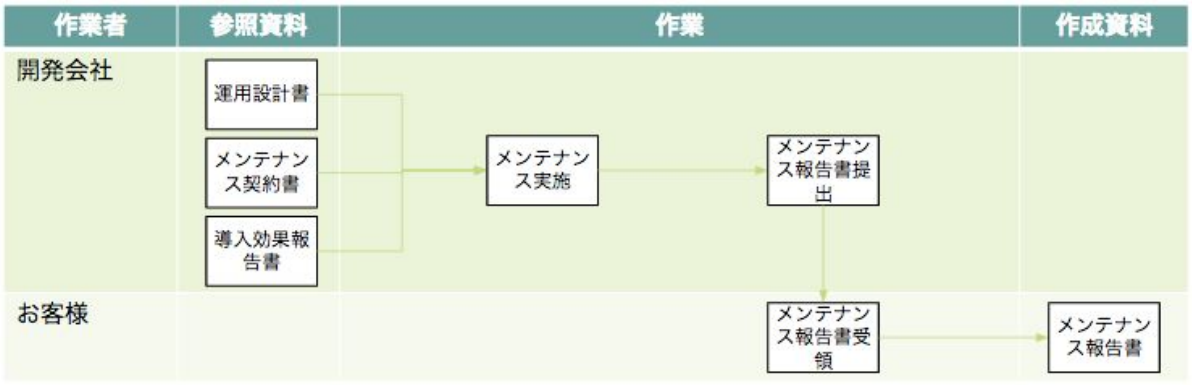
〈主な作業〉
- 運用設計書、メンテナンス契約書に従ってメンテナンスを実施する。
- 定期的（例：月次）にメンテナンス状況の報告書をお客様へ提出する。
- 導入効果報告書を踏まえ、シナリオ設計の修正を検討・提案する。

5.2.2. 作業者と入出力確認表

項目	説明

1	作業者	開発会社
2	参照資料	運用設計書、メンテナンス契約書、導入効果報告書
3	作成資料	メンテナンス報告書
4	完了基準	お客様によるメンテナンス報告書の受領
5	後続プロセス	なし

5.2.3. 作業の依存関係（メンテナンス）



5.2.4. 作業詳細

5.2.4.1. メンテナンス実施

メンテナンス契約に従い、メンテナンスを実施する。
（例：問い合わせ対応、サーバ監視、障害対応、メンテナンス報告書作成・提出等）
また、導入効果報告書を基に、シナリオ設計の見直しを検討する。修正が必要である場合は、修正を提案する。

修正後の配信タイミング

ロボアプリ配信方法は以下となる。

〈ロボアプリ配信方法〉

		Choregrapheから直接配信	マイアプリ配信	安全性審査済のロボアプリ配信
1	ロボアプリ安全性	保証なし	保証なし	保証あり（SBR）
2	リリースまでの時間	即時	即時	最短で2週間
3	配信基盤の有無	なし	あり	あり
4	対象	ローカルネットワーク内のPepper	管理対象のSBRアカウントのみ	全てのPepper for Bizもしくは任意のSBRアカウント（複数選択可能）

5	配信条件	なし	なし	バージョン毎にロボアプリ安全性審査の申請必要

ロボアプリを修正する際には、計画通りに配信するためにロボアプリ安全性審査期間を考慮したスケジュールを立てる。

確認事項（メンテナンス）		確認
1	メンテナンス契約書に従い、お客様に定期的に報告を行っているか	
2	導入効果報告書を基に、シナリオ設計の見直しを提案したか	

5.2.4.2. メンテナンス報告書提出

作成したメンテナンス報告書を、お客様に提出する。

5.2.4.3. メンテナンス報告書受領

メンテナンス報告書を、お客様が受領する。

おわりに

本ガイドラインは社内有識者や社外開発会社様の協力を得ながら作成された。開発会社はそれぞれ独自開発フレームを築いているため、それらを標準化として1つにまとめあげることは容易ではなかった。しかし皆が団結し納得するレベルまで持っていけたことは成果であり、特にいただいた声で一番多かったのは、「企画の段階で効果測定（KGI・KPI）を意識した提案ができていなかったので参考になった」という評価だった。

ロボアプリ開発の特徴はPepperとユーザのインタラクションに関して完成後のアプリを見て初めて細かい要件が決められていく点にある。例えば、単体テスト前にデモアプリを準備し、顧客からフィードバックをもらった上で再度、要件定義書を書き直すという繰り返しのプロセスがこれまで常態化してきた。

この標準ガイドラインではお客様にもロボアプリ開発の全体像を理解していただき、早期にモックなどを活用しゴールのイメージを共有することが短納期・低コスト化実現への近道となると考えている。さらに5つの各フェーズごとにお客様による承認プロセスを設けることで手戻りのリスクを減らす工夫も必要である。

本ガイドラインはロボアプリの1ライフサイクルプロセスについて、当たり前のことが書かれている。ただしソフトバンクロボティクスが自ら最大限のノウハウやヒントを要所要所に盛り込んだ点は品質・技術向上に役立てられるものとして期待したい。これによって各開発会社のロボアプリ開発手法に何ら制限・制約が加わる訳ではない。導入する場合は案件の規模や特性といった市場ニーズの変化に伴って適切にカスタマイズする必要があり、ソフトバンクロボティクス基準という物差しの提示により、各々の優っている点、足りない点をぜひ発見し新たな開発フレームを築いていただきたい。

最後にハードウェアベンダーとしてソフトバンクロボティクスが期待する想いは、Pepperが世の中に対して「人に寄り添うロボット」であり続けられるか、という点である。ぜひこの機会に「Pepperだからこそ実現できること」を再考し市場を盛り上げていってほしい。

用語集

用語	よみがな	定義
SBR	えすびーあーる	ソフトバンクロボティクス（株）のこと
Pepper for Biz	ぺっぱーふぉーびず	法人向けPepperのサービスの総称。ハード、ソフト、ロボアプリ、サービス、全てを含む
お仕事かんたん生成	おしごとかんたんせいせい	お仕事をカスタマイズするWebページ。SBRアカウント（あるいはアルデバランアカウント）によりログインし、設定するできる。サービス名称と、Web名称は同様
Choregraphe	これぐらふ	ソフトバンクロボティクスが開発したロボアプリ開発ツールのこと

参考リンク一覧

カテゴリ	内容	URL
Python	開発ガイド	https://docs.python.org/devguide/
	Python公式サイトのコーディング規約	https://www.python.org/dev/peps/pep-0008/
	Python公式サイトのコーディング規約の日本語訳	http://pep8-ja.readthedocs.io/ja/latest/
NAOqi OS	API ドュキュメント	http://doc.aldebaran.com/
	既知バグ一覧	http://doc.aldebaran.com/2-4/news/known_issues.html
	Developer Portal	https://developer.softbankrobotics.com/
ロボアプリ品質チェックリスト	開発チェックリスト 旧称『マイアプリ開発ガイドライン』	http://cdn.softbank.jp/mobile/set/common/pdf/static/robot/support/document/pepper_myapp_guide.pdf
	解説	http://cdn.softbank.jp/mobile/set/common/pdf/static/robot/support/document/pepper_myapp_guide02.pdf
Pepper for Biz	クラウドサービス	http://www.softbank.jp/robot/biz/cloud-service/
	リリース情報	http://www.softbank.jp/robot/biz/releasenotes/
	サポート	http://www.softbank.jp/robot/biz/support/
	マニュアル情報	http://www.softbank.jp/robot/biz/support/document/
	安全上のご注意	http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-01.htm
	使用上のご注意	http://help.mb.softbank.jp/robot/pepper-for-biz/pc/02-02.html
	Pepperパートナープログラム	http://www.softbank.jp/robot/developer/dev-support/program/partner/
	ロボアプリ安全性審査	http://www.softbank.jp/robot/developer/dev-support/program/partner/benefit/#03
IPA	脆弱性対策	https://www.ipa.go.jp/security/vuln/index.html

免責事項

本書を発行するにあたって、内容に誤りのないようできる限りの注意を払いましたが、本書の内容を適用した結果生じたこと、また、適用できなかった結果について、著者、発行人、発行会社は一切の責任を負いませんので、ご了承ください。

本書の一部あるいは全部について、著者、発行人、発行会社の許諾を得ずに無断で転載、複写複製することは禁じられています。

本書に記載した情報に関する正誤や追加情報がある場合は、Developer Portalサイトに掲載します。以下URLのサイトをご参照ください。

SoftBank Robotics Developer Portal
https://developer.softbankrobotics.com/jp-ja/home

商標

本書に記載する会社名、製品名等は、各社の商標又は登録商標です。

本書の文集は、これらの表記において商標登録表示、その他の商標表示を省略しています。