DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

予算、スケジュール

予算、スケジュール

1. DX企画書に必ず記載する「予算」「スケジュール」
    - いくらかかるのか、いつまでにやるのか
2. ポイント
    - 厳密な予算、スケジュールまでは、企画書では明確に書きにくいケースもある（開発の詳細が決まってから）
    - **だからといって、まったく予算、スケジュールを考えないと、プロジェクトをやるべきか判断ができない**
    - だいたいの予算、スケジュールのイメージをつかむ

予算、スケジュール

具体的なタスクが決まっていないので、
予算もわかりません！
スケジュールもわかりません！

うーん、それだと、プロジェクトを本当に
進めていいのか判断できない・・・

漫画：南 智恵

予算、スケジュール

だいたいの予算、スケジュールを調べる

他の事例	似たような過去のプロジェクトの実績
ヒアリング	過去のプロジェクト経験者に話を聞く
見積もり	外注する場合、費用感を外注先に聞いてみる

予算、スケジュール

外注先にヒアリングしたところ、大体、
このシステム開発なら2～3ヶ月、
500万円くらいでできそうです！

なるほど、思ったよりコストはかからない
ようなので、引き続きプロジェクト設計を
進めよう！

漫画：南 智恵

今回のポイント

予算、スケジュール

（大体）

予算、スケジュール

外注先にヒアリングしたところ、大体、
このシステム開発なら2～3ヶ月、
500万円くらいでできそうです！

なるほど、思ったよりコストはかからない
ようなので、引き続きプロジェクト設計を
進めよう！

漫画：南 智恵

予算、スケジュール

1. 注意点
    - 大体の予算、スケジュールを調べるときには、
      「これは、大体の数字です」と明確に伝える
2. そうしないと
    - 上司から「あれ、500万円で出来るって言ったよね？」
      と言われてしまう
    - **詳細の予算、スケジュールは、具体的に細かいタスクが決まってからじゃないと分からない**

今回のポイント

予算、スケジュール

（大体）

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

計画は「フェーズ」で分ける

計画は「フェーズ」で区切る

1. たとえば
    - 「3年後に、社内のDXをすべて完了させます！」

      → どうやって実現するのか、プロセスが分からない
2. フェーズで区切る
    - 「はじめの6ヶ月で、社内の一部社員向けにテストします」
    - 「次の6ヶ月で改善して、その後、1年で社内に展開します」
    - 「3年後には全社員がデジタルを使うことを目標にします」

      → 期間を区切って説明することが大事

今回のポイント

DXの事例 (1)

DXの事例

1. 経費精算

- レシートに書かれている金額を入力して、経費として会社に申告

  → レシートをカメラで読み取ると、自動的に金額が申告される

計画は「フェーズ」で区切る

1. スケジュール例

- 1年目はテスト
- 2年目から、社内全員が利用可能

	1年目	2年目～
フェーズ	テスト	社内全員が利用可能

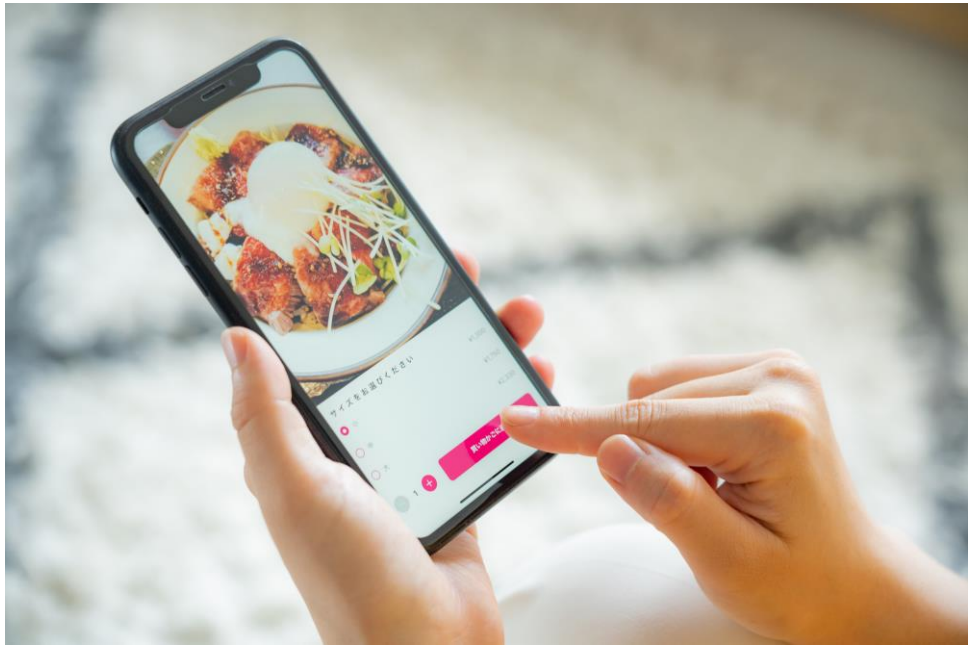
今回のポイント

DXの事例 (2)

注文のDX

1. 効率的な注文システム

- 口頭で注文を受けるのではなく、
- すべてオンラインで注文を受ける



計画は「フェーズ」で区切る

1. スケジュール例

- 1年目は1店舗だけでテスト
- 2年目から、デジタル化が進んでいる東京都内で展開

  → 多くのお客様がデジタルを使ってくれたら、
- 3年目から全国展開

	1年目	2年目～	3年目～
フェーズ	1店舗	都内10店舗	全国100店舗

今回のポイント

計画は「フェーズ」で分ける

計画は「フェーズ」で区切る

1. たとえば
    - 「3年後に、社内のDXをすべて完了させます！」

      → どうやって実現するのか、プロセスが分からない
2. フェーズで区切る
    - **「はじめの6ヶ月で、社内の一部社員向けにテストします」**
    - 「次の6ヶ月で改善して、その後、1年で社内に展開します」
    - 「3年後には全社員がデジタルを使うことを目標にします」

      →「テストフェーズ」などと呼ぶ

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

テスト計画

テスト計画

1. テスト

- DXを導入するときは、必ずテストを行う
- いきなり社内全体に展開すると、トラブルが起きるかも

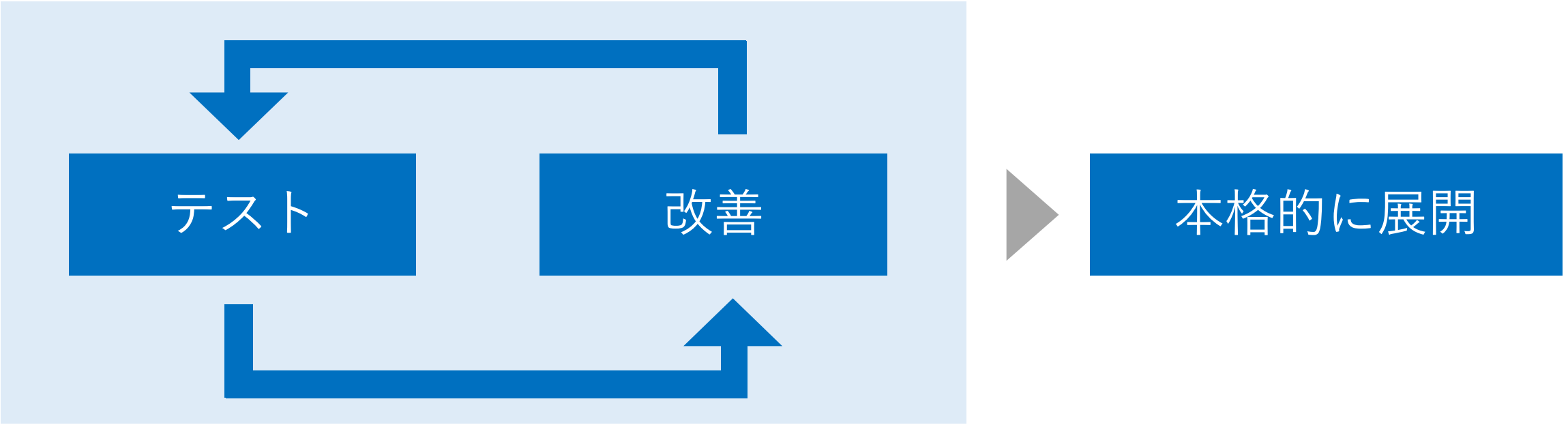
  → ためしに一部の社員だけで、デジタルを使ってみる

テスト

1. テストと改善

- **「これなら本格的に展開しても大丈夫そうだ」**

  と確認できるまで、テストと改善を繰り返す



今回のポイント

テスト（例）

DXの事例

1. 経費精算

- レシートに書かれている金額を入力して、経費として会社に申告

  → レシートをカメラで読み取ると、自動的に金額が申告される

テスト

1. 経費精算のDX
    - ためしに、100枚のレシートと領収書をカメラで読み込む

      → きちんと正しい金額を読み取れているか確認する
2. 改善の例
    - うまく読み込むときのコツや、
    - なかなか読み込めない種類のレシートがないか

      → マニュアルにして、本格展開するときに説明する

      → トラブルを少なくすることが可能

今回のポイント

テスト（例）

コールセンターのDX

1. コールセンター

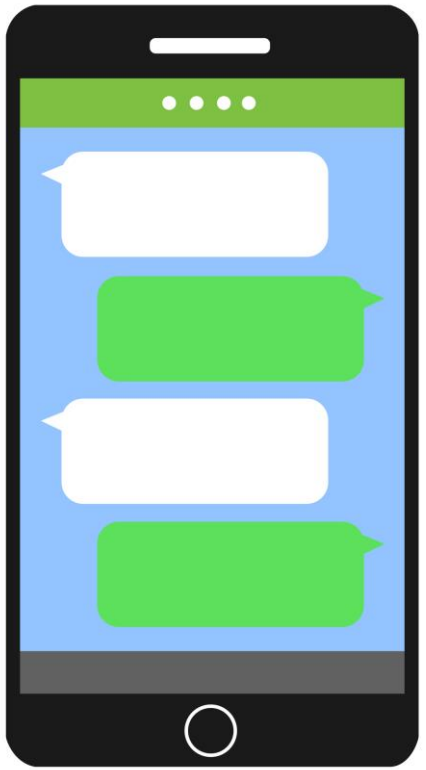
- 顧客からの問い合わせを電話で対応

→ DX化したい・・・どうする？

コールセンターのDX

2. チャットで質問すれば自動で回答してもらえるシステム

- **まずはチャットで入力・相談すると、自動的に教えてくれる**
- それでも分からないところがあれば電話で相談
- 顧客も待ち時間が少ない
- 電話の件数も減るので、社員の負担も減る



テスト

1. コールセンターのDX
    - ためしに、一部の顧客に対してチャット機能を提供する

2. KPI
    - 顧客がチャット機能に満足したか確認する
        - 「チャット機能に満足しましたか？」
        - 「口頭とチャット機能、どちらが便利と感じましたか？」

→ チャット機能に満足していなければ改善を進める

今回のポイント

テスト計画

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

段階的にリリースする

段階的にリリースする

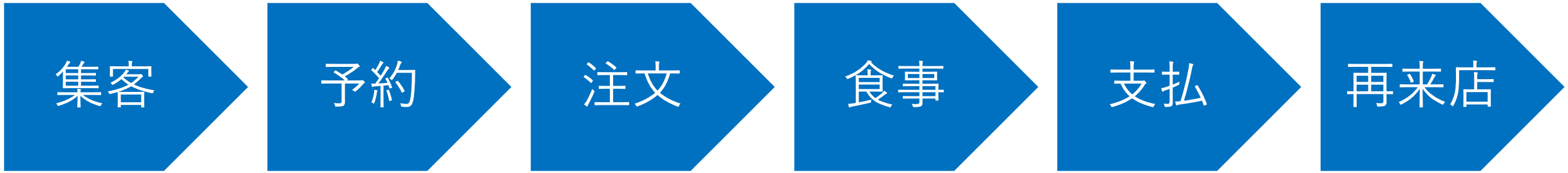
1. 意味
    - いきなり全ての業務をデジタル化するのは大変
    - **優先度の高い業務からデジタル化していく**

2. ポイント
    - 機能を小さく分けて、すばやくリリースしていく

飲食店のDX

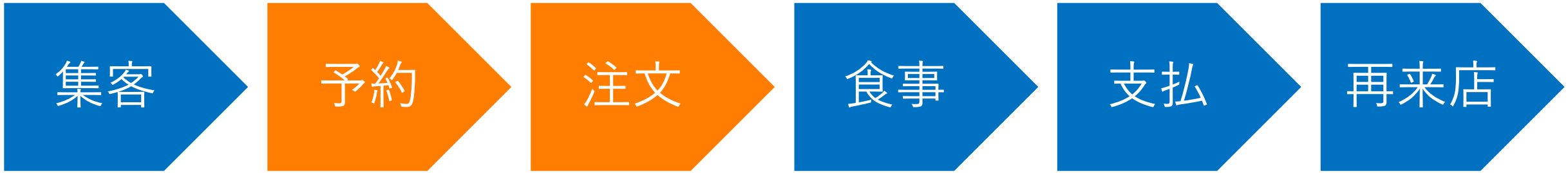
飲食店のDX

1. 段階的にリリースする



飲食店のDX

1. 段階的にリリースする
    - まずは、予約と注文をデジタル化しよう

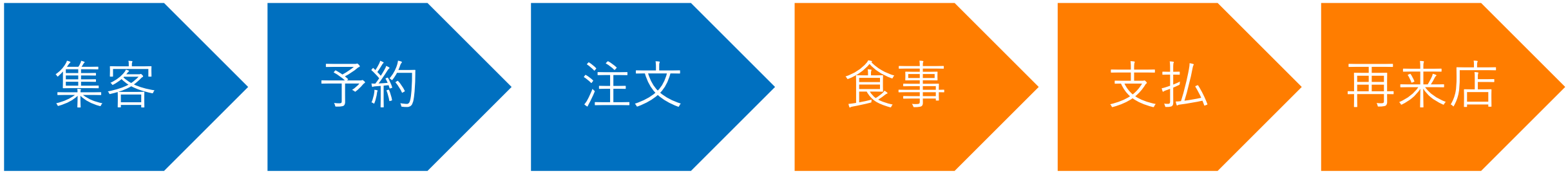


飲食店のDX

1. 段階的にリリースする

- まずは、予約と注文をデジタル化しよう
- **ユーザーが予約と注文をシステムで使ってくれることを確認**

  → それ以外もデジタル化しよう

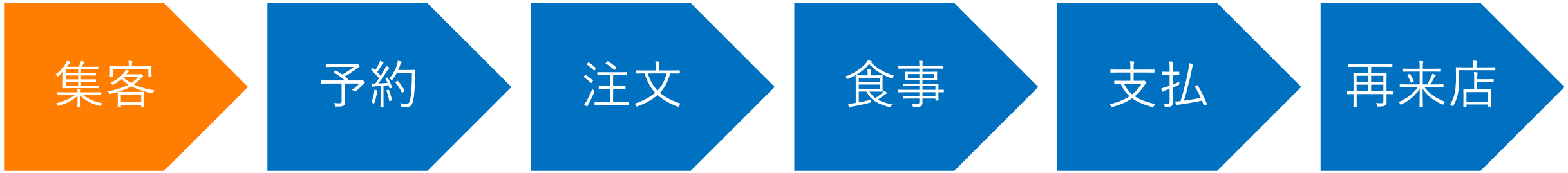


飲食店のDX

1. 段階的にリリースする

- まずは、予約と注文をデジタル化しよう
- ユーザーが予約と注文をシステムで使ってくれることを確認

  → それ以外もデジタル化しよう
- **うまくいったら、集客もデジタル化して来客数を増やそう**



今回のポイント

アジャイル

アジャイル

1. 組織でよくある問題
    - プロジェクトを進めるときに、
    - 企画、設計、開発・・・各プロセスに時間がかかりすぎる
    - せっかく企画・設計までまとめても、開発で問題が発生してしまい、また企画からやり直し・・・
2. ポイント
    - アジャイル型の進め方を検討してみる
    - Agile ＝ 機敏な

アジャイル

ウォーターフォール

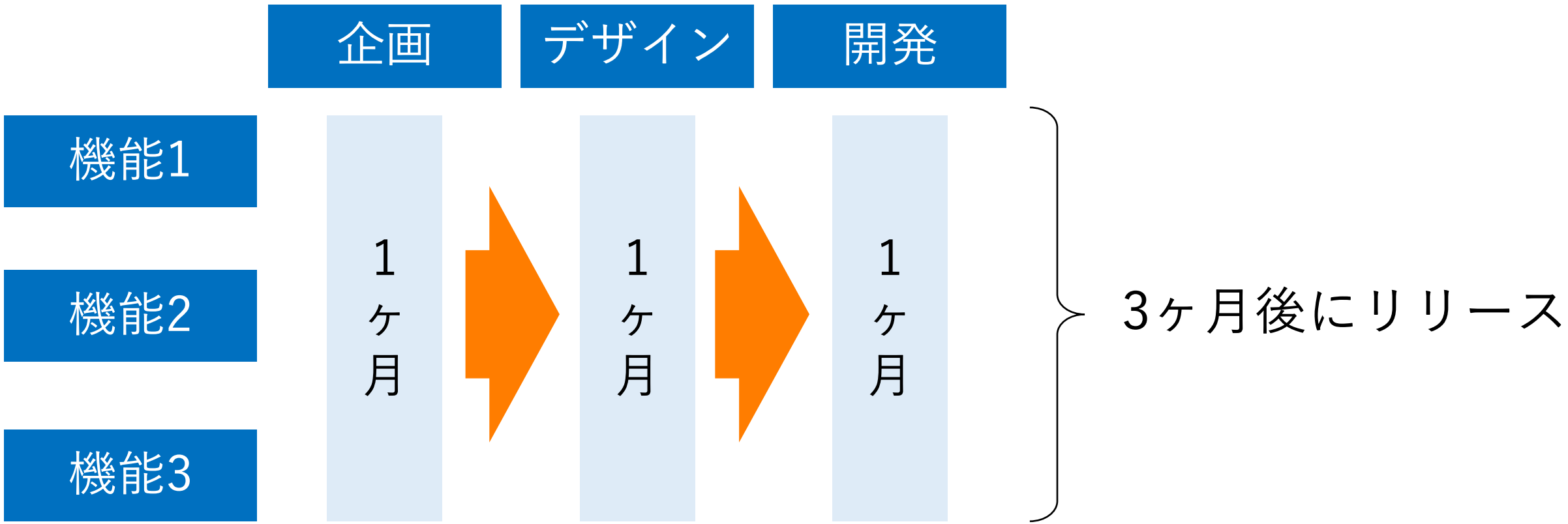
- すべての機能を、まとめて企画→開発
- 時間がかかる、修正しにくい

アジャイル

- 機能を分けて、細かく企画→開発
- 時間をかけず、修正しやすい

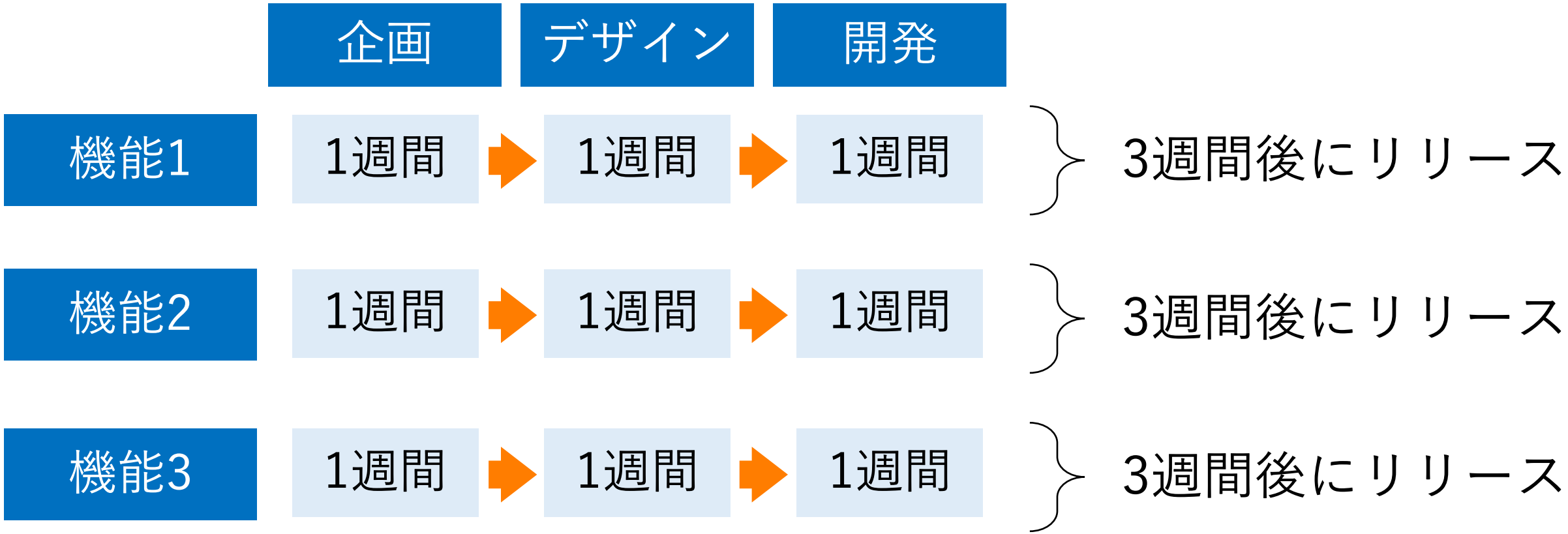
アジャイル

ウォーターフォール



アジャイル

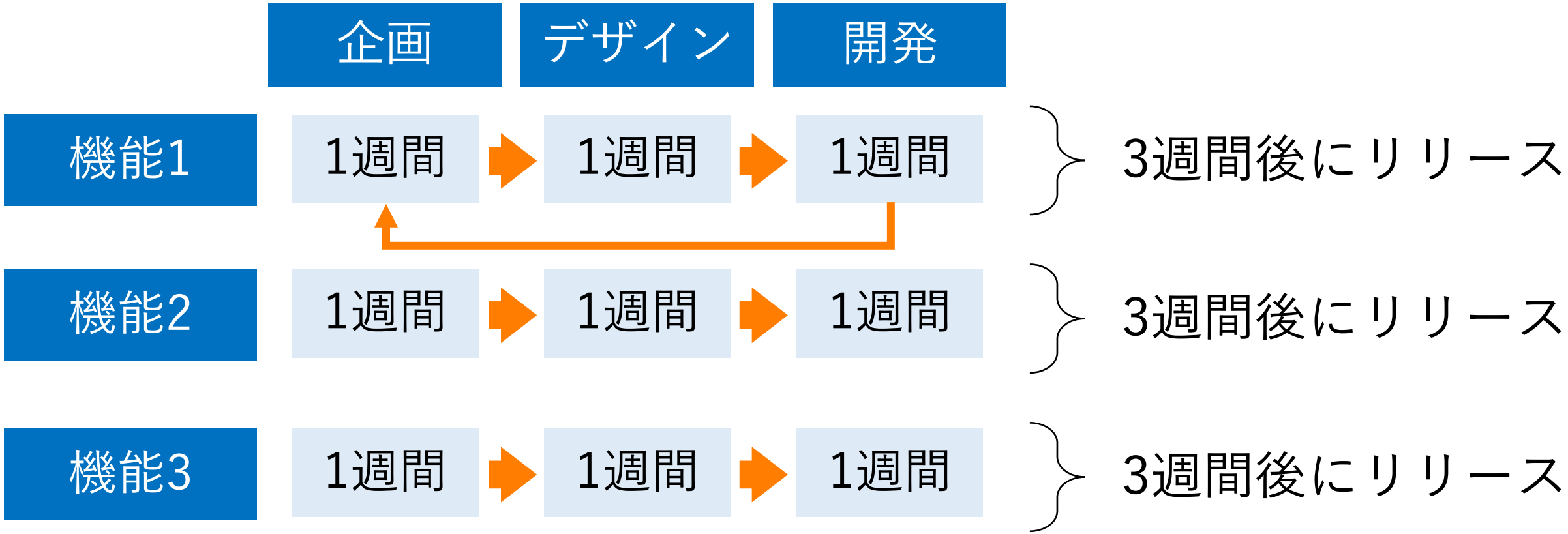
アジャイル


企画
デザイン
開発
機能1
1週間
1週間
1週間
3週間後にリリース
機能2
1週間
1週間
1週間
3週間後にリリース
機能3
1週間
1週間
1週間
3週間後にリリース

アジャイル

開発で問題が発生しても、企画に戻りやすい

→ プロジェクトを修正しやすい



アジャイル

ウォーターフォール

- すべての機能を、まとめて企画→開発
- 時間がかかる、修正しにくい

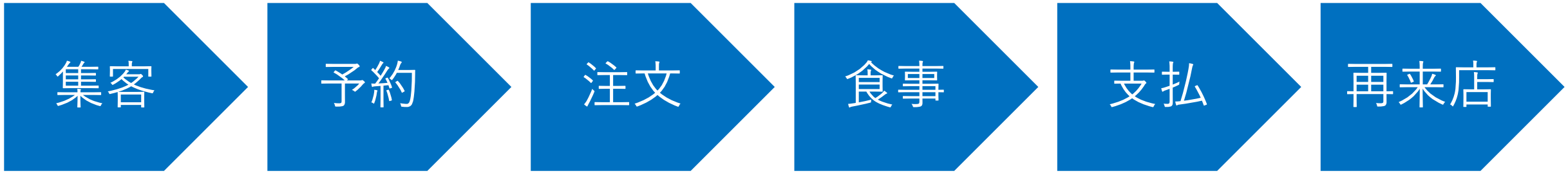
アジャイル

- 機能を分けて、細かく企画→開発
- 時間をかけず、修正しやすい

飲食店のDX

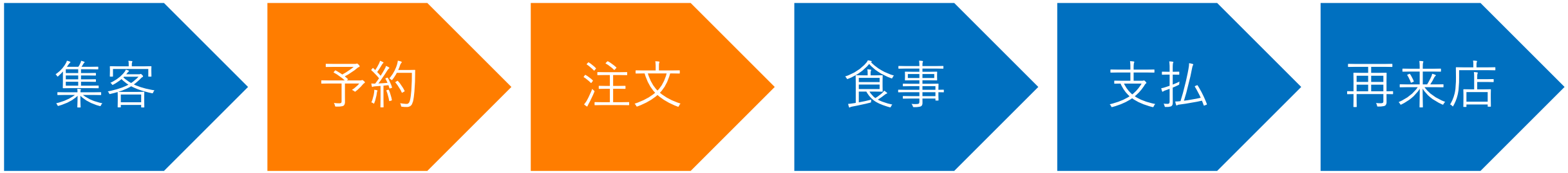
飲食店のDX

1. アジャイル開発



飲食店のDX

1. アジャイル開発
    - まずは、予約と注文をデジタル化しよう

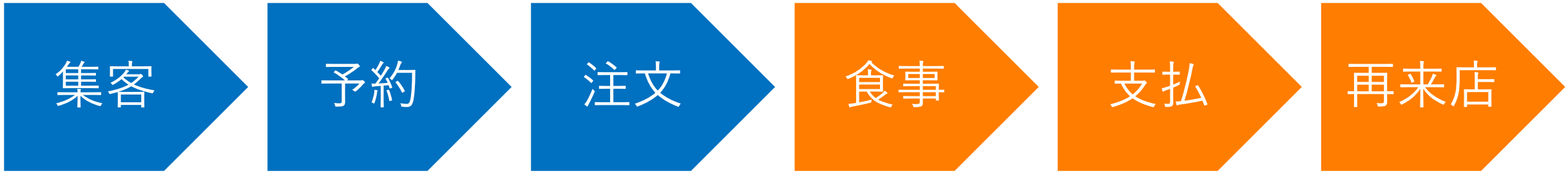


飲食店のDX

1. アジャイル開発

- まずは、予約と注文をデジタル化しよう
- **ユーザーが予約と注文をシステムで使ってくれることを確認**

  → それ以外もデジタル化しよう

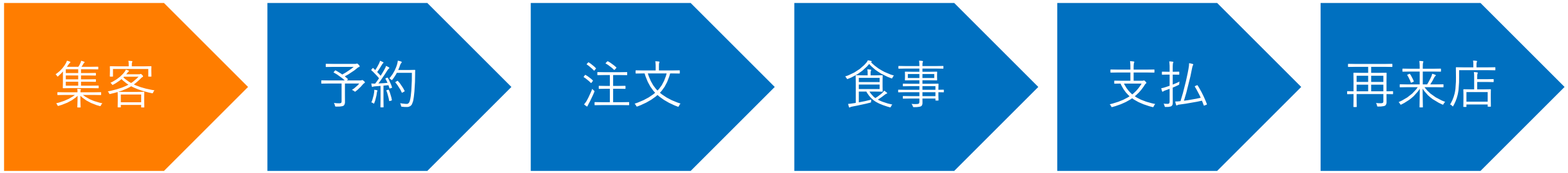


飲食店のDX

1. アジャイル開発

- まずは、予約と注文をデジタル化しよう
- ユーザーが予約と注文をシステムで使ってくれることを確認

  → それ以外もデジタル化しよう
- **うまくいったら、集客もデジタル化して来客数を増やそう**



今回のポイント

アジャイル

今回のポイント

小さく、速く、始める

DXプロジェクト

1. ポイント
    - **とにかく重要なのが「スピード」**
        - はやく始めて、問題があればすぐに改善していく
    - **デジタルサービスは修正が簡単**
        - だからこそ、完璧を目指すよりも、
          すぐにサービスを開始して改善を続ける

今回のポイント

Done is better than perfect

（完璧を目指すより、まず終わらせろ）

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

データの準備期間

1. ポイント

- DXを進めるときには、データの準備が必要
- スケジュールを考えるときに、データの準備期間も加える

今回のポイント

データの準備

DXで重大な判断ミスを防ぐ

医師による医療画像の検査、診断

- 画像検査の見落とし
- 疾患を見つけるべきところを、

  見落としてしまった

  → 医療ミス（人為的ミス）

  → 患者側から訴訟されるケースも
- **医療画像を分析して疾患を**

  見つけるシステム

DXで品質を維持する

1. 工場

- 製品に問題がないか、人間の目でチェック

  → 製品をカメラで分析して、その画像から、

  不具合がありそうなところを発見する

データの収集

1. データ分析には、正解データ（教師データ）が必要
    - 医療画像（疾患ありの画像、なしの画像）
    - 不良品の画像

    → 正しいデータと答えを大量に用意しないと、

    推測の精度も上がらない

今回のポイント

教師データ（例）

繰り返し作業

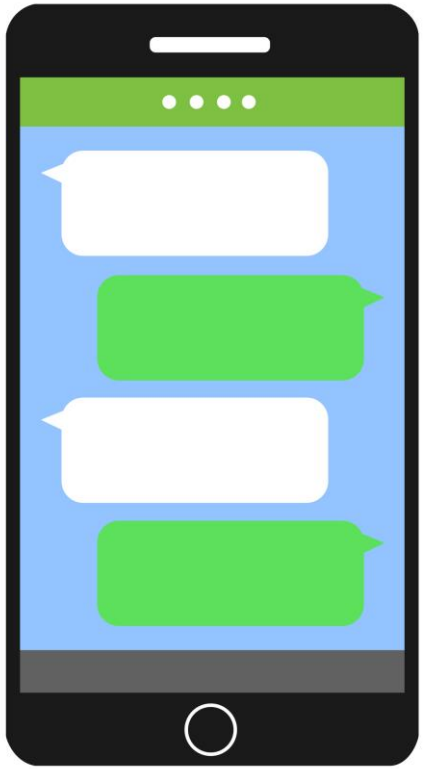
1. 同じような質問をたくさん受ける

- 社員「うちの会社の退職金制度について質問があります」
- **人事「また同じ質問か・・・毎日毎日・・・面倒だな・・・」**

繰り返し作業

1. 同じような質問をたくさん受ける

- 社員「うちの会社の退職金制度について質問が・・・」
- **まずはチャットで入力・相談すると、**
  自動的に教えてくれる
- それでも分からないところがあれば
  人事に直接相談
  → 人事の「繰り返し作業」を効率化できる



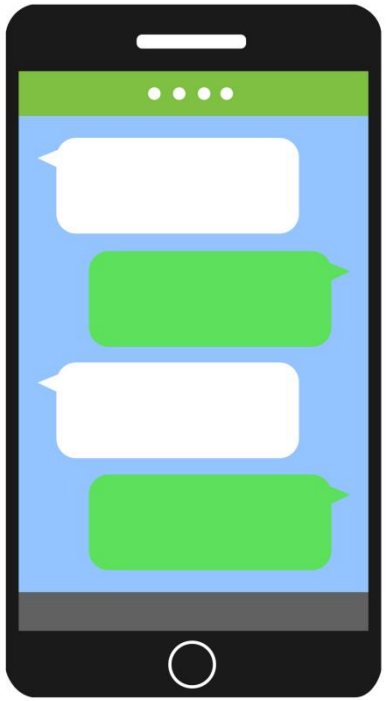
今回のポイント

必要な教師データは？

繰り返し作業

1. 教師データ（例）

- 想定される質問と、その回答をテキストで用意する
- 過去の会話（質問、回答）の音声データを用意する



今回のポイント

データの準備期間

データの準備期間

1. ポイント

- DXを進めるときには、データの準備が必要
- スケジュールを考えるときに、データの準備期間も加える

データの準備期間

1. スケジュール例

- 3ヶ月かけてデータを集める
- その後9ヶ月で開発、テスト
- 2年目から社内全員が利用可能

	3ヶ月	9ヶ月	2年目～
フェーズ	データ準備	開発・テスト	社内展開

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

DXプロジェクトのリスク管理

デジタル化のリスク管理

1. リスク管理

- DXによって発生する「新たなリスク」を企画書に記載する
- そのリスク管理が十分であることをチームで確認する

→ 確認しないと、トラブルが起きたときに企画者1人に
責任を押し付けられる可能性も

2. リスク（例）

- 法律　　違法行為にあたらないか
- 個人情報　　意味なく個人情報を収集していないか

今回のポイント

違法行為

リスク管理

1. 著作権、肖像権

- さまざまな画像データを利用する際、

  その画像の著作権を侵害していないか？

2. 法律

- 新しいデジタル技術の活用が、法律上、問題ないか？
- オンライン診療のケース

  （昔は、オンライン診療は認められていなかった）

今回のポイント

個人情報

データの収集と個人情報

1. データ収集で気をつけたい点

- 外部のユーザーから多くのデータを集めれば、
  マーケティングなどに活用できる
- **しかし、ユーザーから収集するデータが、**
  「万が一、流出してしまったら？」
  を想定すると、個人情報をたくさん収集するのは危険

データの収集と個人情報

1. ウェブサイトでユーザーが登録する情報（例）

- どちらのほうが情報漏洩したときのリスクが高いか？

ウェブサイトA	vs	ウェブサイトB
• ID		• 実名
• ニックネーム		• 年齢、性別
• 年齢		• 住所、電話番号

データの収集と個人情報

1. ウェブサイトでユーザーが登録する情報（例）

- どちらのほうが情報漏洩したときのリスクが高いか？

→ Bの情報が漏洩したら、ユーザー個人が特定されてしまい、住所、電話番号など個人情報が広まってしまう

ウェブサイトA	vs	ウェブサイトB
• ID		• 実名
• ニックネーム		• 年齢、性別
• 年齢		• 住所、電話番号

DXの実装計画

1. 予算、スケジュール
2. 計画は「フェーズ」で区切る
3. テスト計画
4. 段階的なリリース
5. データの準備期間
6. リスク管理
7. プロジェクトメンバー
8. まとめ

今回のポイント

プロジェクトメンバー

プロジェクトメンバー（関係者）

1. 経営陣
    - 会社の予算をDXプロジェクトに投資してくれる責任者は？
2. 開発部門
    - 技術的に開発は可能か？
3. 法務
    - 法務面の問題はないか？（データの個人情報、著作権など）
4. 営業部門、顧客
    - 彼らが使いこなせるシステムか？

今回のポイント

他にも

オンライン契約書

プロジェクトメンバー

1. 契約書のDX

- 弁護士、法務部門の協力が不可欠

画像認識のビジネス活用(1) 医療

医師による医療画像の検査、診断

- 画像検査の見落とし
- 疾患を見つけるべきところを、

  見落としてしまった

  → 医療ミス（人為的ミス）

  → 患者側から訴訟されるケースも
- **医療画像を分析して疾患を**

  見つけるシステム

プロジェクトメンバー

1. 医療画像の検査のDX
    - 医師の協力が不可欠
    - もしAIによる検査結果にミスがあった場合、どうする？
        - 法務の意見も重視

今回のポイント

プロジェクトメンバー