Wagby Developer Day 2023

スムーズなローコード開発を目指して！ Wagby導入時に確認するべきポイント、開発を促進するテクニック

2023年10月26日

株式会社ソフトウェア・パートナー

自己紹介

株式会社 ソフトウェア・パートナー

金子　英俊　（Hidetoshi Kaneko）

- 2007年から「Wagby専任技術者」として
  システム開発と技術支援に従事
- Wagbyエキスパート認定技術者
- 趣味はB級映画鑑賞（おすすめはサメ系）

今回のテーマ

ローコード開発で苦労しない為に、事前に確認すべきポイント、メンテナンスやテストを考慮した設計時の考え方について

AGENDA

- （入門）ローコード開発のメリットを生かすには
- （実装/モデリング）使いやすさと保守性のバランスについて
- （個人経験談）アジャイル開発との相乗効果

セッションのターゲット層

- 対象者
- ①これからWagby（ローコード開発）の利用を検討されている方
- ②ここ最近始められた方
- ③なんとなく興味がある方

ローコード開発の
メリットを生かすには

ローコード開発のメリットを生かすには

- Wagbyを含むローコード開発の導入でよくある問題に現行踏襲があります。

成功

・現行システムの正確な仕様書
・現行システムの理解度
・ストアドプロシージャなど資源の流用

失敗

・過剰な現行踏襲
・開発ツールの仕様違いへの不理解
・現行システムの不理解

ローコード開発のメリットを生かすには

- Wagbyは標準機能による対応とカスタマイズによる対応で実現可能な範囲が異なります。

標準機能

・モデリングから画面のレイアウト調整まで開発ツール上で設定

- **開発期間の短縮**
- **開発コストの削減**
- **開発の自由度に制限有り**

カスタマイズ

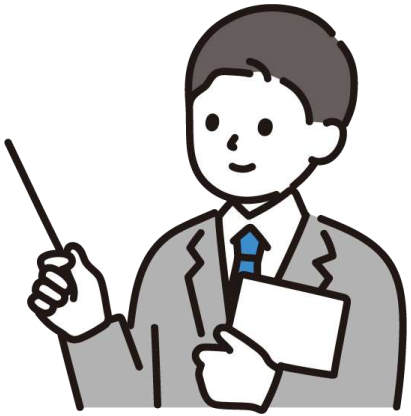
・自動生成されたソースコードを更にカスタマイズ

- **開発期間の延長**
- **開発コストの増加**
- **標準仕様を超える挙動に対応**

ローコード開発のメリットを生かすには

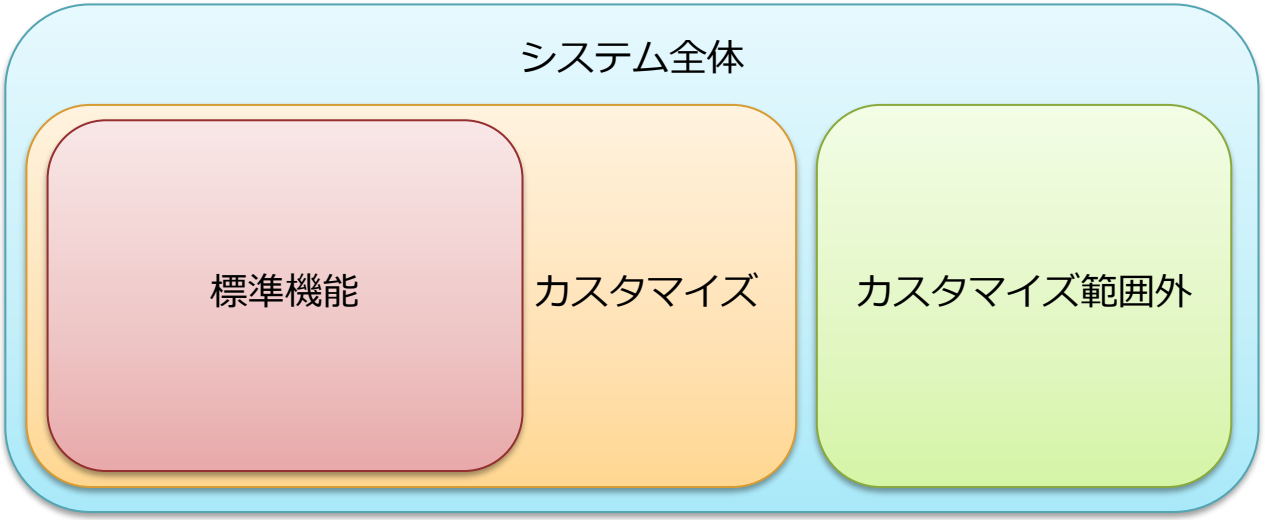
過剰なカスタマイズはローコード開発の
メリットを低下させる

Wagbyの標準機能で処理を実現する程、
ローコード開発のメリットが最大化する！



ローコード開発のメリットを生かすには

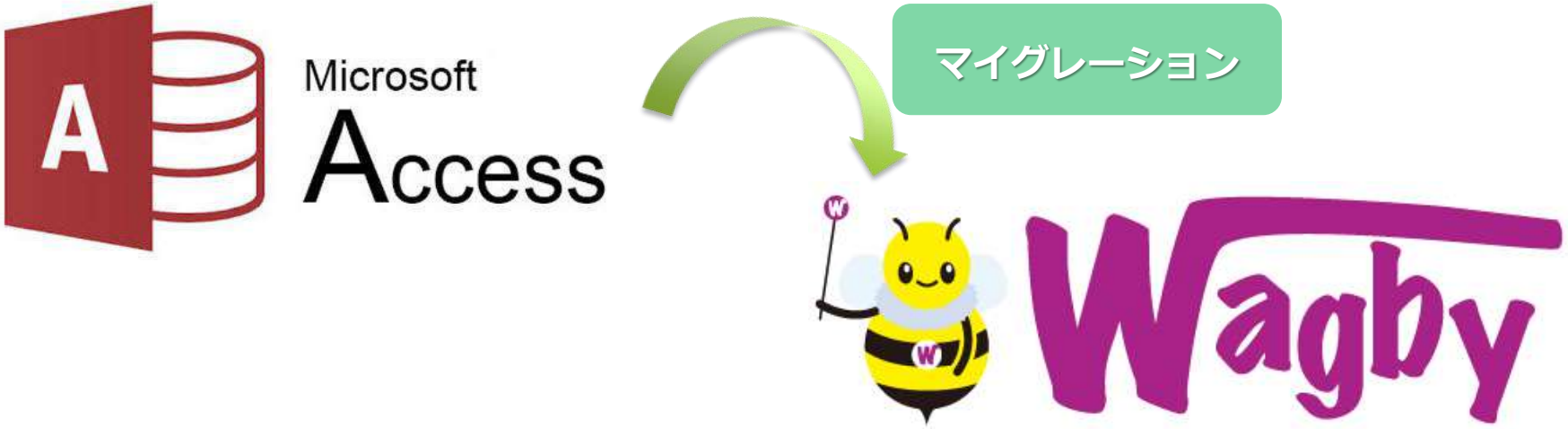
- カスタマイズ前提の設計を行う前に標準機能での実現可能な範囲を把握する。



メーカーサイト参照：https://wagby.com/wdn9/index.html

ローコード開発のメリットを生かすには

- Accessで作成された業務アプリをWagby環境へマイグレーションする場合



ローコード開発のメリットを生かすには

- Wagbyの機能を最大限に利用して、作業を効率化

Wagbyリポジトリを半自動生成

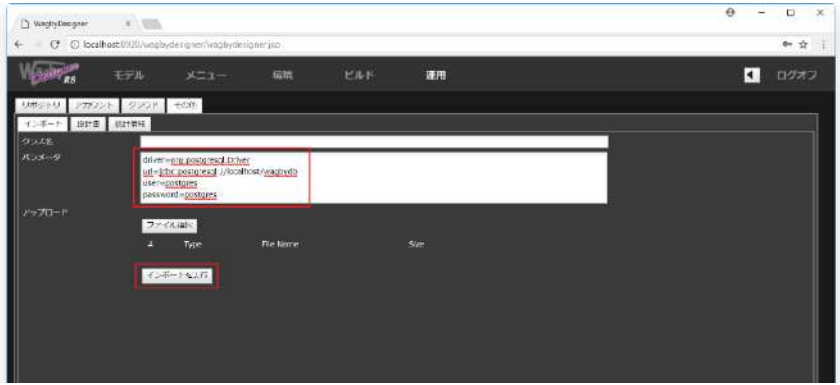
既存システムのリレーショナルデータベースからテーブル情報を取得し、
Wabgyのリポジトリを生成することで、モデル作成の作業を軽減し、設定ミスを防止。

サブデータベースの利用

別システムのデータベースからマスタ情報を共有し、マスタデータの同期作業を不要に。

ローコード開発のメリットを生かすには

- 既存テーブル定義からリポジトリを生成する
- https://wagby.com/manual8/tools-readschema.html



ローコード開発のメリットを生かすには

現行システムの資源を利用

集計用のビューテーブルやストアドプロシージャなどを、そのまま利用可能。
これらの資源は既に利用実績があり、詳細なテストを行う必要が無い為、
工数の削減に寄与。

Wagbyに合わせたレイアウト調整

入力欄などは可能な限り、現行システムに近づけてレイアウト、ボタン配置などは
Wagbyの制限に合わせた調整を行い、レイアウト関連のカスタマイズを避ける。

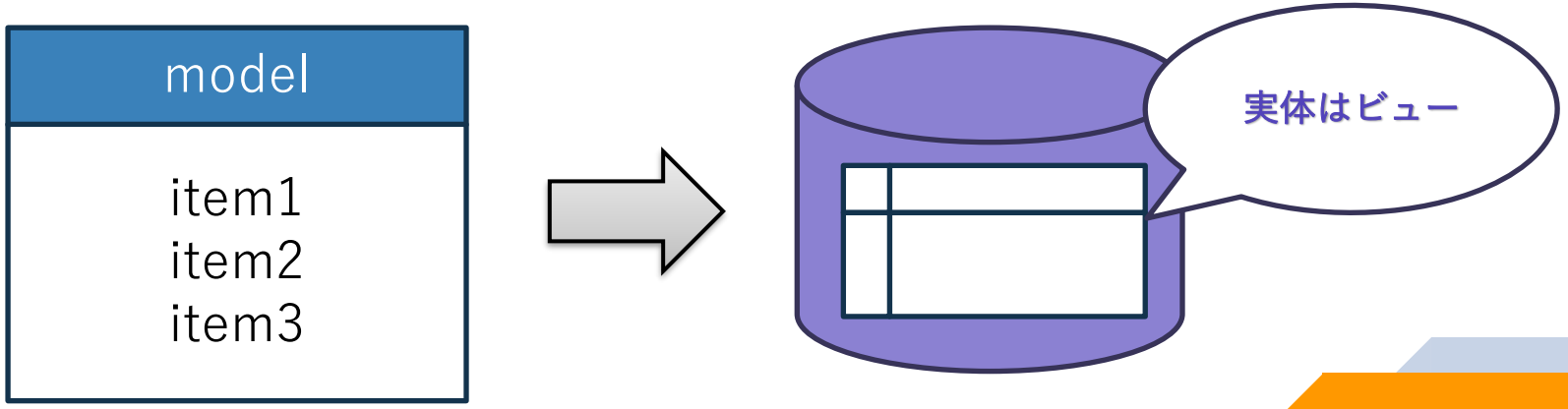
メリットを生かす心得としてまずは・・・

- ローコード開発ツールは、なんでもできる魔法の道具ではないと理解する。

- 使う者が開発ツールの仕様と長所に合わせた設計・実装理念を持つことがすごく重要！

ローコード開発のメリットを生かすには

- データベースのビューを利用する
- https://wagby.com/manual9/model-useview.html

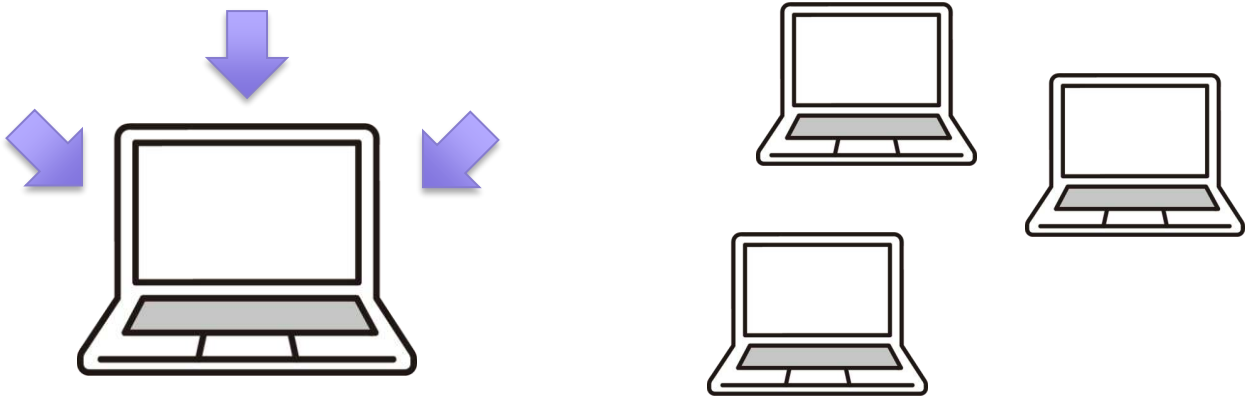


議題 2

使いやすさと保守性の
バランスについて

使いやすさと保守性のバランスについて

- Wagbyで社内システムなどを設計する際に
  使いやすさを考慮して、１つの画面に出来るだけ機能を集約する考え方と
  保守性を考慮して、特定権限のユーザのみ使用する機能やレアケースを
  対応する機能などを別画面に分散する考え方があります。



使いやすさと保守性のバランスについて

- モデリングとは
- https://wagby.com/tutorial9/howtodesign1.html

設計情報を構成する要素

使いやすさと保守性のバランスについて

使いやすさを優先しすぎたモデリング(機能の詰込み型)

メリット

・操作が分かりやすく、ユーザの使い勝手が良い。

・必要なモデル数が少なく出来る。

・画面を行き来しない為、保存するフラグなどを少なく出来る。

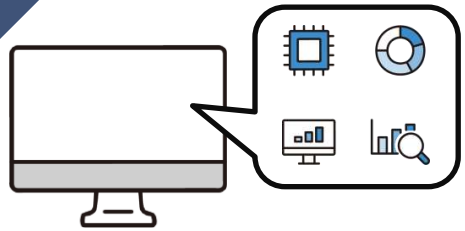
デメリット

・内部の処理が複雑化する為、管理コストが高くなる。

・機能を追加する際に影響範囲が広く、テストケースが増える。

・１モデル内の項目数が極端に多くなる為、管理に手間がかかる。

使いやすさと保守性のバランスについて



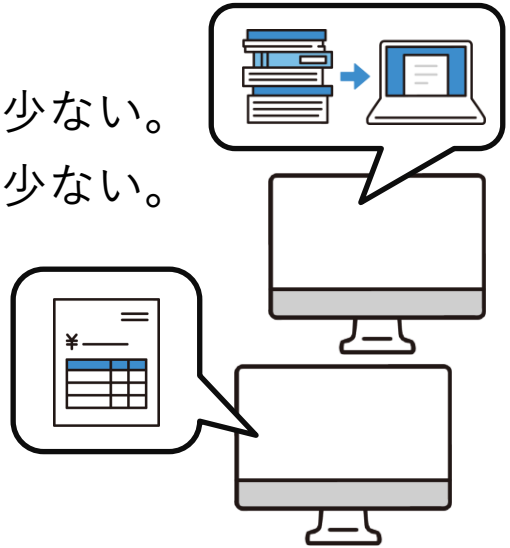
使いやすさだけを優先せずに保守性を考えたモデリング（機能の役割分担型）

メリット

・機能追加を行う際の影響範囲が狭いので、テストケースが少ない。
・障害が発生しても部分的なものになり、他機能への影響が少ない。
・各処理が軽く管理コストが低く抑えられる。

デメリット

・操作が分かり難くなる為、ユーザの使い勝手は悪くなる。
・サブモデルを多用する為、モデルの総数が増えてしまう。
・画面遷移などのUI関係に工夫が必要になる。



使いやすさと保守性のバランスについて

・簡単に設定可能なUI方面の工夫

利用する機能に合わせて画面遷移させる。

使いやすさと保守性のバランスについて

分かり難い操作がある場合、該当箇所にヘルプ機能を用意する。

使いやすさと保守性のバランスについて

機能の系統別にボタンの背景色を分ける

・例）全画面で帳票出力を行うボタンの背景色を黄色に統一する。

使いやすさと保守性のバランスについて

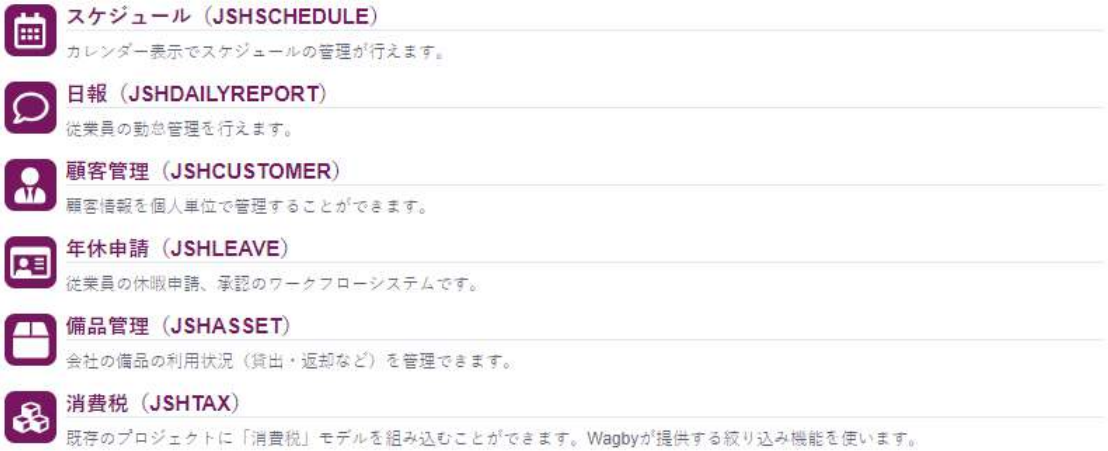
・長期運用を想定したシステムをWagbyで開発する場合、
保守性があまりに悪いとレガシーシステムとなってしまい、
メンテナンスにかかるコストの増大を招きます。

・ユーザの使いやすさを優先しつつも、可能な範囲で機能を分散させるなど
保守性の確保も考慮した設計が必要です。

・Wagbyには画面遷移やヘルプメッセージなどのUIを補助する機能も
豊富なので、保守性を優先して低下した使いやすさを補うことも可能です。

使いやすさと保守性のバランスについて

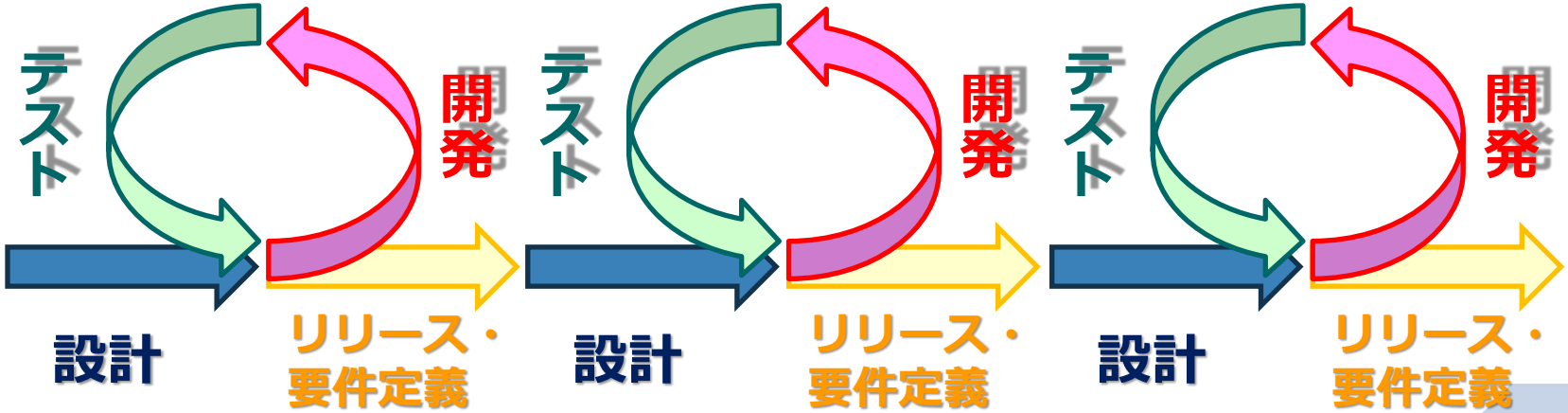
- モデル構成のサンプルとして
  アドオンギャラリーを参考にする
- https://wagby.com/tutorial9/index.html



アジャイル開発との相乗効果

アジャイル開発との相乗効果

- ノーコード・ローコード開発は、開発工程の小さなサイクルを繰り返すアジャイル開発との親和性が高い為、合わせて用いられる事が多く、Wagbyでの開発もアジャイル開発で行われることが増えています。



アジャイル開発との相乗効果

アジャイル開発のメリット

・開発スピードが早い

・１サイクル毎に柔軟な仕様変更が可能

・ユーザーのニーズを反映しやすい

ローコード開発（Wagby）のメリット

・開発期間やコストが抑えられ、生産性が高い

・プログラミングスキルなしでも一定の品質でシステムを構築可能

・既存システムとの連携が可能

・セキュリティ対策の負担を軽減

アジャイル開発との相乗効果

アジャイル開発のデメリット

・全体のスケジュールのコントロールが難しい

・コスト管理が難しい

・頻繁な仕様変更がドキュメントに反映され辛い

ローコード開発（Wagby）のデメリット

・開発の自由度はツールに依存している

・動的コンテンツや複雑な要件のシステム開発には不向き

・開発ツールによる直感的な開発が行われると
仕様・設計のドキュメントが作成・更新されない

アジャイル開発との相乗効果

・アジャイル開発に向いたプロジェクトとは

- 要件や仕様が未確定のプロジェクトの場合
- 仕様変更や機能追加が多く予想されるプロジェクトの場合
- クライアントが参画する度合の高いプロジェクトの場合

まとめ

Wagbyの標準設定での対応範囲、カスタマイズでの対応範囲を把握して、効率的に設計しましょう。

アプリの長期利用を想定している場合は、使いやすさと保守性のバランスも考慮しましょう。

Wagbyはトライ＆エラーを繰り返すアジャイル開発にも向いている為、機能拡張を繰り返す開発にも対応可能です。

ソフトウェア会社の役割は、システムを構築することではなく、

顧客の問題を解決することである！

株式会社ソフトウェア・パートナー