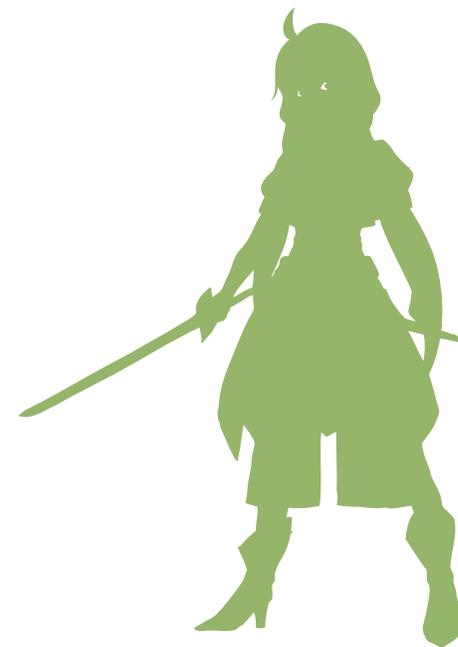


# 依存性を排除したデータベース アプリケーション構築実践法

第35回エンバカデロ・デベロッパークャンプ

エンバカデロ・テクノロジーズ セールスコンサルタント  
筑木真志



**e**mbarcadero®  
DEVELOPER CAMP

# アジェンダ

- OracleからSQL Serverに移行した事例のお話
- 移行作業における問題点の考察とその解決
- まとめ

# OracleからSQL Serverに移行した事例のお話

- その1：当時の状況と結果



## とある現場で、このようなお願いをされました



お宅のシステムに興味あるのだけど、うちのSQL Serverで動かないかな...

# 当時の状況 その1 : システム条件

- おおむねこんな感じの良くある組み合わせです

項目	製品
OS	Windows Server
データベース	Oracle + SQL Server
O/R マッパー	MyBatis
Webサーバー	IIS
アプリケーションサーバー	Apache Tomcat + Apache Struts 2 + JSP
フロントエンド	jQuery UI

## 当時の状況 その2 : データベース設計方針

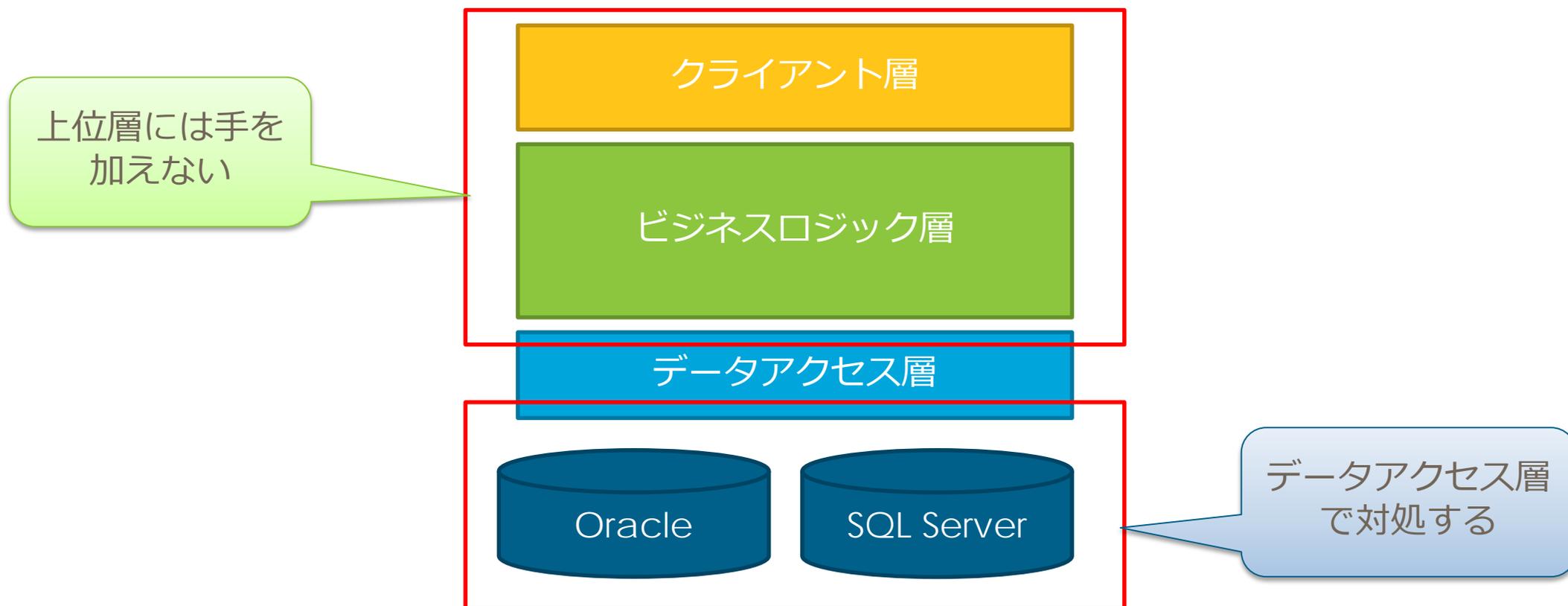
- RDBはOracle XEを想定
- 外部整合性やトリガーは使用しない
- RDBはデータの保持に徹する
- 複雑なクエリや、汎用性のあるクエリはVIEWや表関数にする

## 当時の状況 その3 : データベースの管理

- テーブル定義書は、所謂Excel設計書
- マクロでテーブル定義書からDDLを生成する
- 論理名を入力すると対応するフィールドの物理名と型・サイズを別の列に出力する
- ビューはSQL文をそのまま記載する

# 移行のストラテジー

- クライアント層とビジネスロジック層は変えない



## その結果...

- 問題なく移行できました！



# でも...

- 結構苦勞もしました



# OracleからSQL Serverに移行した事例のお話

- その2：移行作業の簡単な概要



# OracleとSQL Serverの主な違い

	Oracle	SQL Server
構造	インスタンス/スキーマ/テーブル	インスタンス/データベース/スキーマ/テーブル
標準の認証方法	データベース認証	Windows認証、SQL Server認証
スキーマ	スキーマとユーザーが同一	スキーマとユーザーが分離
文字列型	VARCHAR2/CHAR	NVARCHAR/NCHAR
文字コード	UTF-8（日本語は主に3バイト）	UTF-16（日本語は主に2バイト）
文字列リテラルの表現	‘文字列’	“文字列”
文字列結合	/ CONCAT関数 ※引数 2 個	+ / CONCAT関数 ※引数可変
NULL値判定	NVL(c, v)/NVL2(c, v1, v2)	ISNULL
日付型	DATE/TIMESTAMP	SMALLDATETIME/DATETIME
型変換	to_char/to_number/to_dateなど	convert
DUAL表	あり	なし

# 移植に当たって その0 : 事前準備

- O/RマッパーはMyBatisを使用していたので、MapperファイルをOracle用とSQL Server用に分離
- mybatis-config.xmlをOracle用とSQL Server用に作成し、ターゲット毎に切り替える

# 移植に当たって その1 : データ型の差異の吸収

- 一般的なデータ型しか使っていなかったため機械的に置き換えた
- DDLはテーブル定義書から生成しているため、フィールド単位の差異はそこで吸収

Oracle	SQL Server	備考
VARCHAR2(n)	NVARCHAR(n/3)	原則として1/3で切り上げ電話番号など日本語を使用しないフィールドについてはVARCHARを使用
CHAR	CHAR	CHAR型はフラグ類しか使わないルールだった
NUMBER(n,m)	DECIMAL(n,m)	
DATETIME	DATETIME2	
BLOB	BLOB	

## 移植に当たって その2 : 互換関数の作成

- OracleにあってSQL Serverにない関数のうち、メジャーかつ、移植可能な物はストアドファンクションを作成
- 参考 : TRIM関数はSQL Server 2017でサポートされた

## 移植に当たって その3 : 複雑なSQL

- 複雑なSQLはVIEWもしくは表関数化
- I/Oを同一にして、中身を入れ替える

# 移植に当たって その4 : ストアドプロシージャ

- 一生懸命移植する
  - SQL Server Migration Assistantの力を借りるのは非常に有効な手段



何が「依存性を排除する」  
だ！！！！

# 移植に当たって その5 : テスト

- 同じデータを与えて、結果が同じかどうかで同一性を担保する
  - 単体テストのテストケースを流用するのがベスト
  - 投入データとその**エビデンス**は「未来の自分のために」 **必ず残す**
- データアクセス層での同一性が担保出来れば、それより**上層のテストは不要**
  - Oracle版とSQL Server版の両方で「総合テスト」やらないよね...
  - やりたいのであれば半々にするとか
  - でも、最低、疎通確認くらいはしようね...

# 移植に当たって その6 : データ移行

- マスタデータは片方を正とし、一方向に変換する
- データの移行は以下の方法を使用
  - BULK INSERT/SQL Loader
  - INSERT文そのもの
  - 各種ツールの使用
    - SQL Developer
    - SQL Server Management Studio
    - SQL Server インポートおよびエクスポート ウィザード
    - SQL Server Migration Assistant
    - A5:SQL Mk-2
    - など

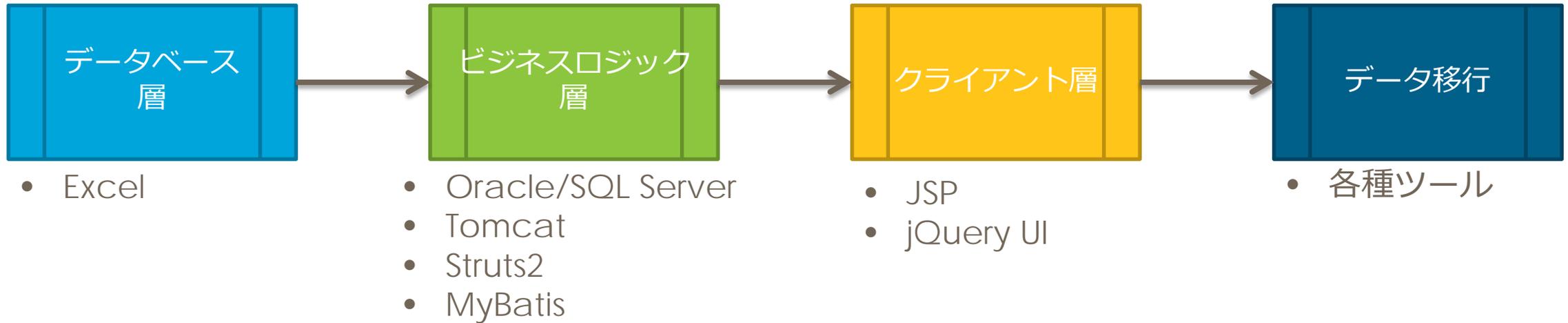
# 移行作業における問題点の考察とその解決

なにが問題だったのか？



**e**mbarcadero<sup>®</sup>  
DEVELOPER CAMP

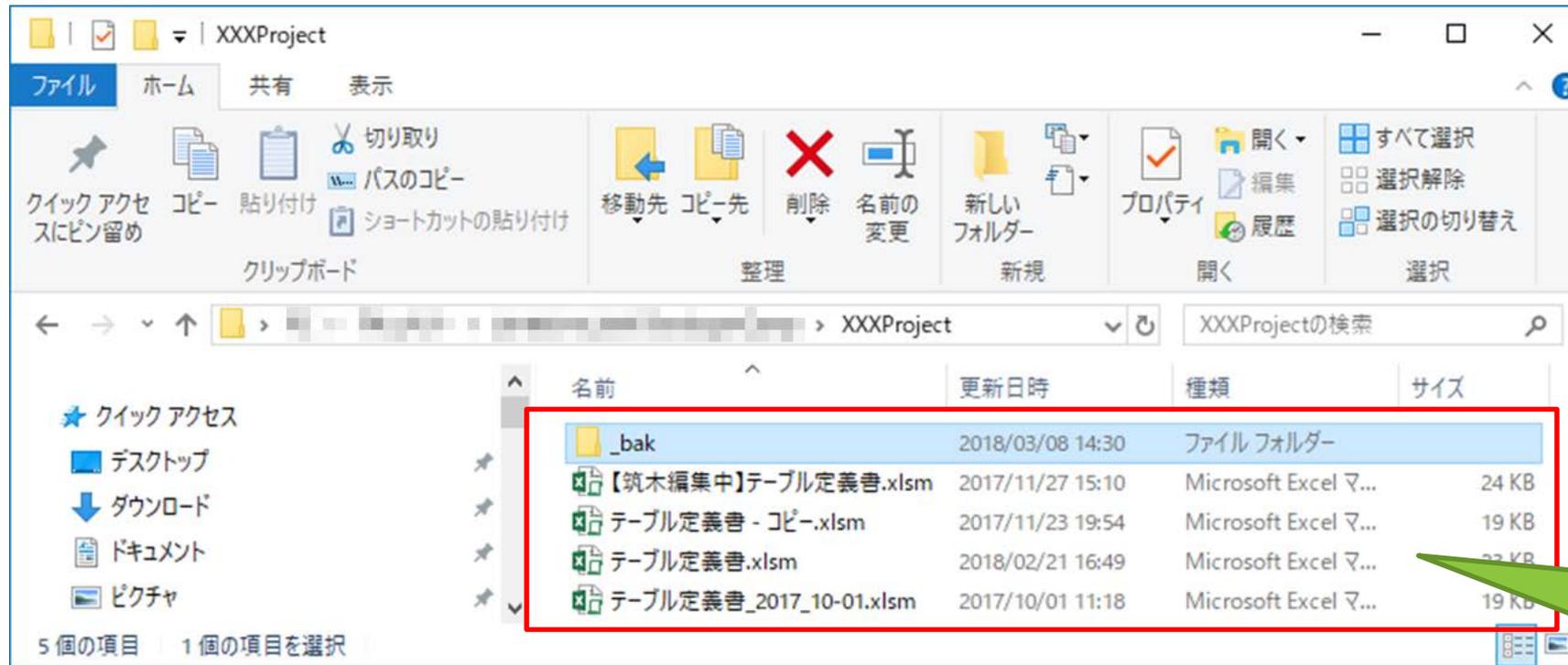
# 移行作業の流れと登場人物の復習



**登場人物、多くありませんか？**

# 問題点その1：Excelブックによるデータベース管理

- Excelなので、誰もが気軽に弄れてしまいます。



## 問題点その2：特定DBに依存したシステム

- 今回に対応するDBが2種類なのでDDLを出力するマクロの修正量  
があまりありませんでした
- MyBatisなどのマッパーファイル、  
メンテナンスし続けられますか？
- 今後、Db2とかPostgreSQLとかMySQLなどに対応する場合は？
  - データベースが増える度にマクロを書き換えますか？
  - そして、そのテーブル仕様書はメンテナンスし続けられますか？

## 問題点その3：曖昧になった三層構造

- StrutsのView層にビジネスロジックが記載されていた
  - View層は画面に関することであって、ビジネスロジックはModel層に記載すべき
- 面倒な画面の実装とそのデバッグ
  - 画面表現がJSP側とjQuery UI側とで分離している
  - **メンテナンスしづらい**ですね？
- この処理の守備範囲はどっち？
  - 入力値のバリデーション
  - あるコードがデータベースにあるかどうか？

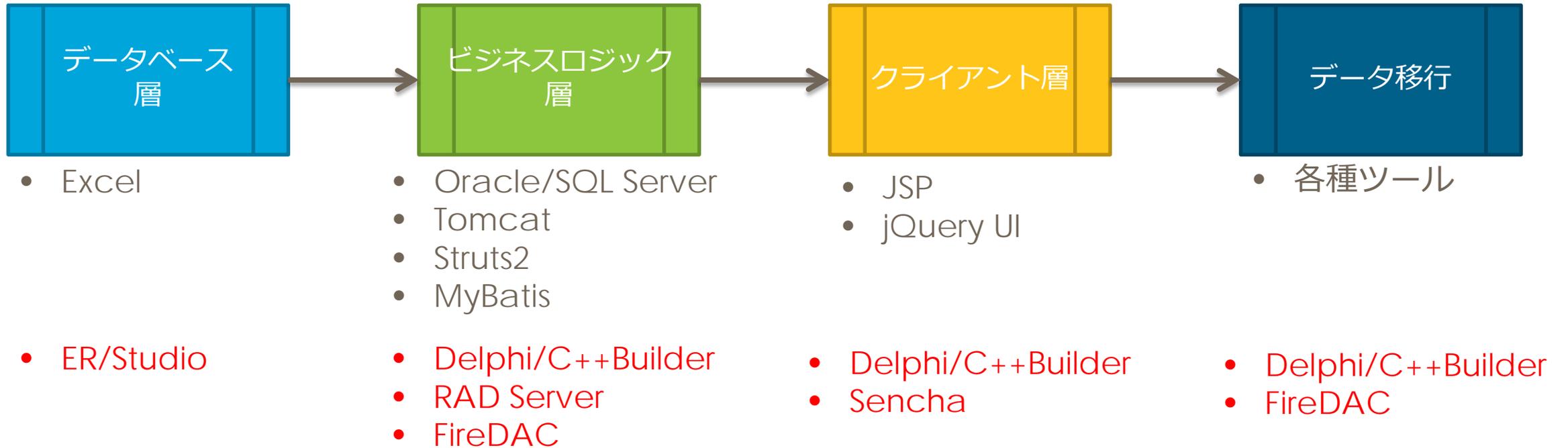
# 移行作業における問題点の考察

エンバカデロのツールを使った解決法



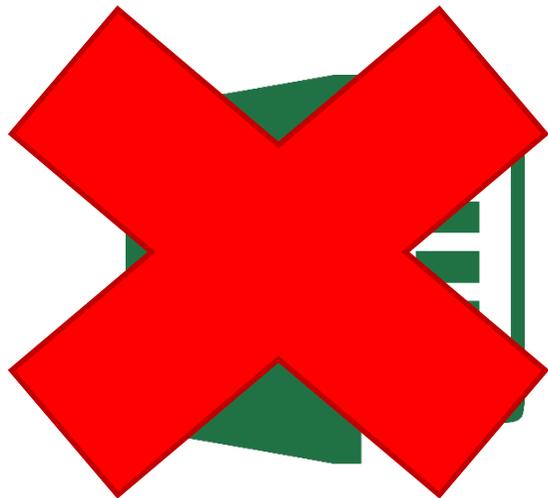
**e**mbarcadero®  
DEVELOPER CAMP

# エンバカデロのツールを使って改善します

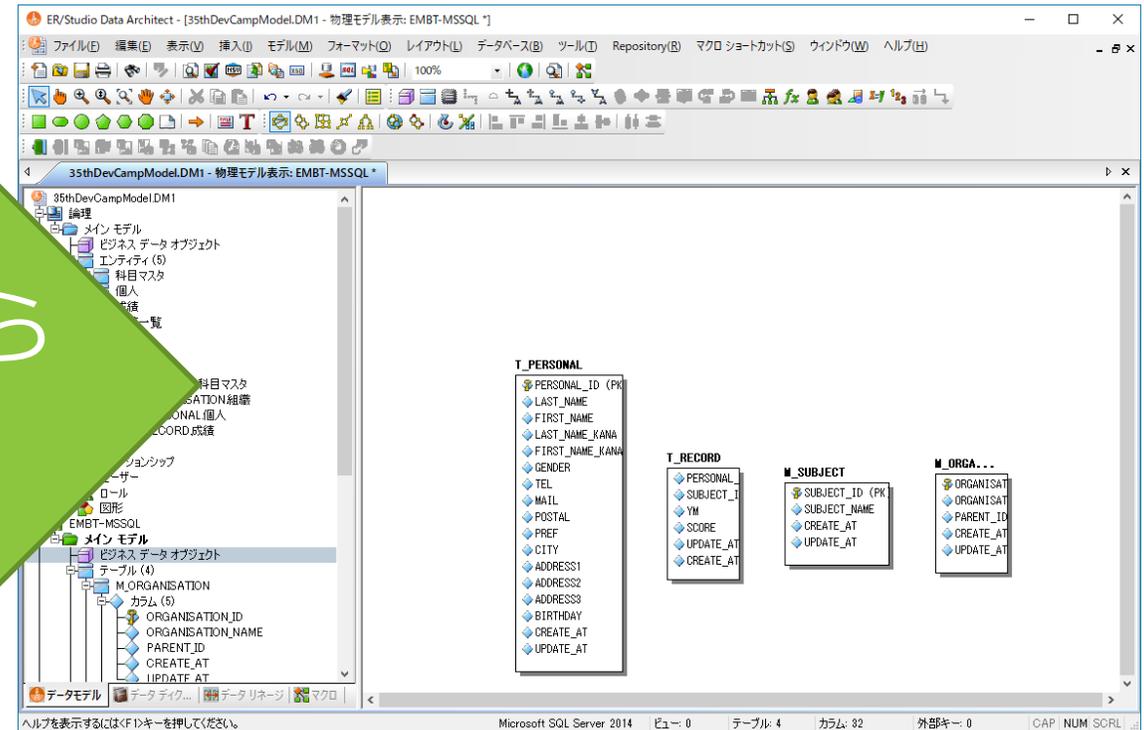


# 改善その1 : ER/Studioでデータベース設計を一元化する

- データのモデリング
- データベースに応じたDDLの生成
- テーブル定義書の生成

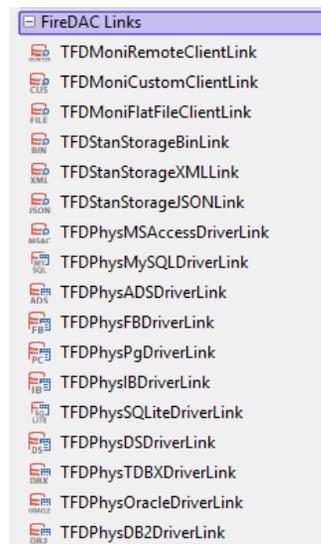
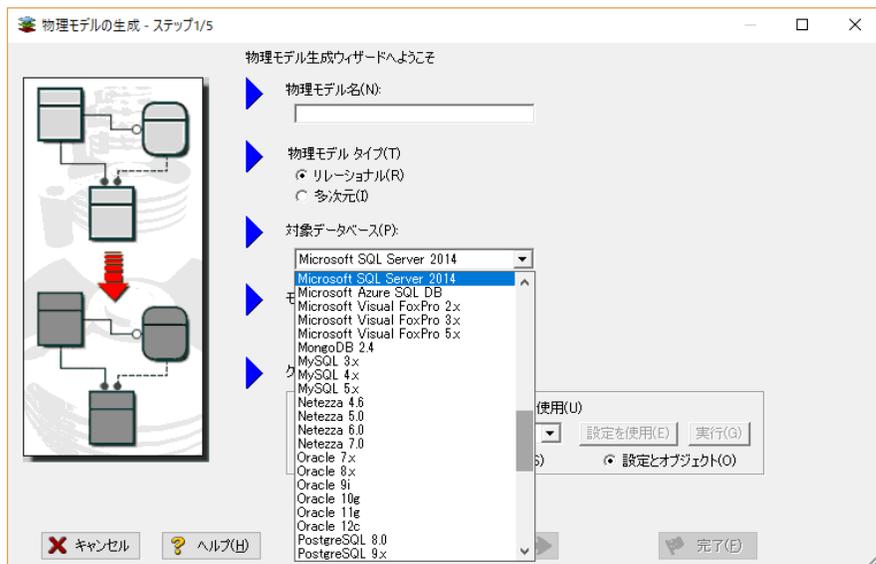


Excel設計書からの脱却！



# 改善その2：特定データベースの依存性解消

- ER/StudioとFireDACは**多種多様**なデータベースに対応
  - 管理：ER/Studio
  - 構築：Delphi/C++Builder & FireDAC
    - FireDACはマクロ機能にてデータベースの**依存性をカバー**出来ます



## 改善その3：システム内の役割の明確化

- システム内の役割分担が明確になった見通しの良いシステム
- フロントエンド、バックエンド共に多彩な選択肢
- レガシーなシステムのモダナイズ

階層	選択肢
クライアント層	Delphi/C++Builder Sencha など
ビジネスロジック層	RAD Server & Delphi/C++Builder
データベース層	Delphi/C++Builder & FireDAC

# まとめ

- KISSの原則
- 単純化する事による、役割分担の明確化
- 昨日の自分は他人。そして、明日の自分も他人
- 移行ツールは移行ツール管理ツールは管理ツールです
  - 餅は餅屋。Excel仕様書で管理するのは止めましょう！
- 残す物は残す。捨てる物は捨てる物の見極めが肝心

# THANKS!

[www.embarcadero.com/jp](http://www.embarcadero.com/jp)

第35回 エンバカデロ・デベロッパーキャンプ