ＶＤＭによる画面機能仕様記述

ＶＤＭの概要

- ＶＤＭは、Vienna Development Methodの略で形式手法のひとつ
  - 数理論理をベースとした仕様記述のための手法
  - 厳密なルールに従った仕様の記述
- 開発者にとって馴染みやすい
  - プログラミング言語に近い書き方
  - 仕様アニメーション（仕様の実行、テスト）
  - 日本語の書籍や無償ツール
- 開発上流工程で利用しやすく、産業界で実績がある

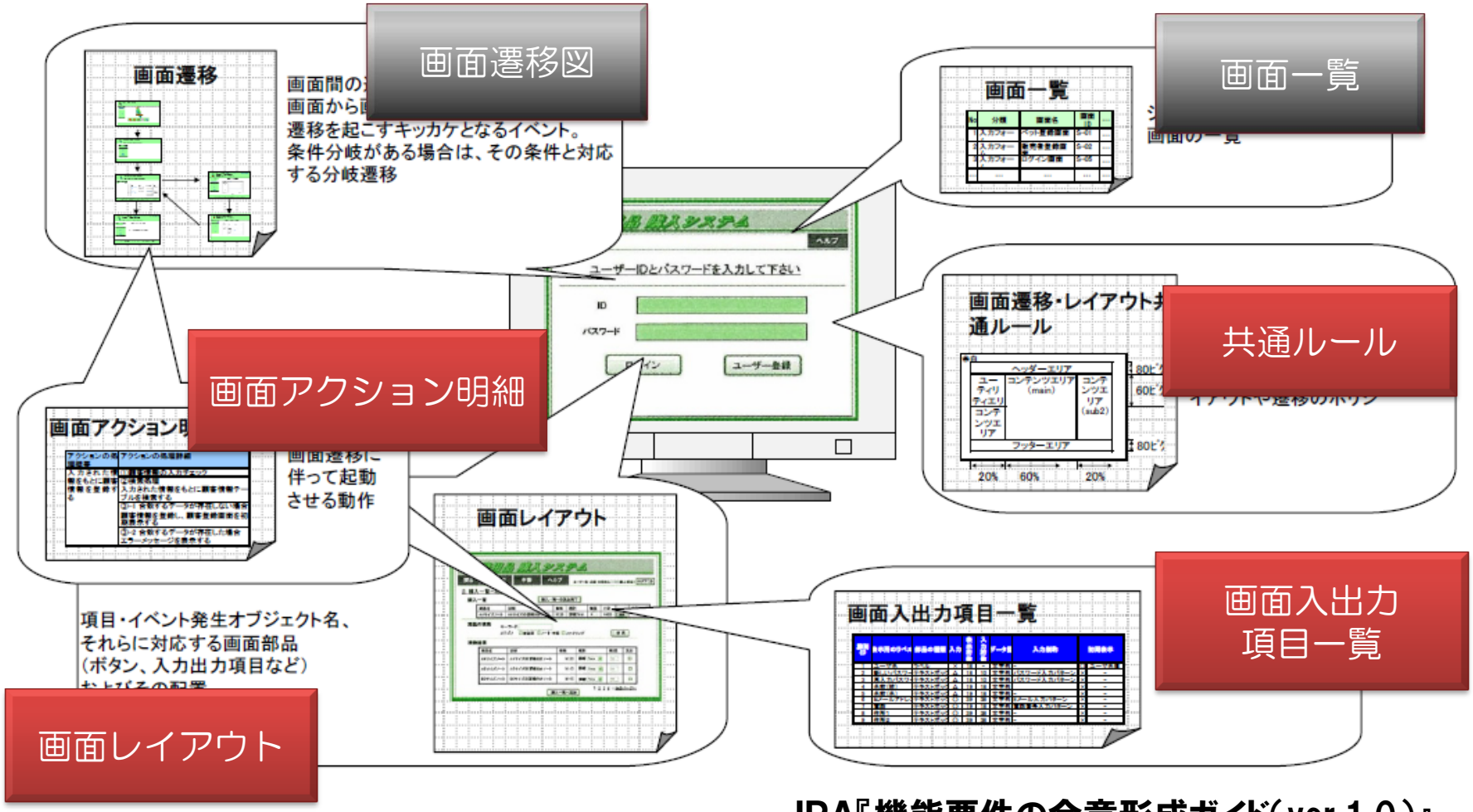
背景と課題

- 背景
  - 社内において、ＷＥＢベースの業務アプリケーションは、開発対象となることの多いシステムの一つである。
- 課題
  - 上記のようなアプリケーションを開発する際に用いられる、「画面機能仕様書」では、画面イメージ（見た目）を扱うことで直感的に伝わる一方、仕様の記述が端折られて曖昧であったり漏れたりし、その結果、誤解も生みやすい

画面機能仕様書に対してＶＤＭを適用することが、課題の解決に役立てられそうかを検討

題材：登録／変更／削除、検索／閲覧といった典型的な機能が備わったサンプルWEBアプリケーション

対象とする画面機能仕様書の構成



IPA『機能要件の合意形成ガイド(ver.1.0)』

VDMの適用（1／3）

- 機能仕様の記述
  - 画面レイアウト、共通ルールをインプットに記述

- ユーザーイベントや引き起こされるアクションの定義

```
-- ================================================
-- Events
      public 検索条件を入力する : 企業名入力 * 企業コード入力 * 業種入力 ==> ()
      検索条件を入力する(a企業名, a企業コード, a業種) ==
        (
            is not yet specified;
        )
        post i検索条件入力 = mk_検索条件入力(a企業名,a企業コード,a業種);

      public 検索ボタンを押下する : () ==> ()
      検索ボタンを押下する() ==
        (
            検索処理(i検索条件入力);
        )
        post (len i企業一覧 > 0 and not i警告表示)
             or (len i企業一覧 = 0 and i警告表示);

-- ================================================
-- Actions
      public 検索処理 : 検索条件入力 ==> 検索結果
      検索処理(a検索条件) ==
        (
            is not yet specified;
        )
        pre not ( a検索条件 = mk_検索条件入力("","",<選択なし>) )
        post card(RESULT) >= 0;
```

VDMの適用（2／3）

- 機能仕様の記述
  - 画面レイアウト、共通ルールをインプットに記述

```
-- ================================================
-- Events
      public 検索条件を入力する : 企業名入力 * 企業コード入力 * 業種入力 ==> ()
      検索条件を入力する(a企業名, a企業コード, a業種) ==
        (
            is not yet specified;
        )
        post i検索条件入力 = mk_検索条件入力(a企業名,a企業コード,a業種);

      public 検索ボタンを押下する : () ==> ()
      検索ボタンを押下する() ==
        (
            検索処理(i検索条件入力);
        )
        post (len i企業一覧 > 0 and not i警告表示)
             or (len i企業一覧 = 0 and i警告表示);
-- ================================================
-- Actions
      public 検索処理 : 検索条件入力 ==> 検索結果
      検索処理(a検索条件) ==
        (
            is not yet specified;
        )
        pre not ( a検索条件 = mk_検索条件入力("","",<選択なし>) )
        post card(RESULT) >= 0;
```

- イベントやアクションが動作する前後に成り立っておくべきことを記述

事前条件
事後条件

VDMの適用（2／3）

- 機能仕様の記述
  - 画面レイアウト、共通ルールをインプットに記述

陰仕様

```
-- ============================================
-- Events
    public 検索条件を入力する : 企業名入力 * 企業コード入力 * 業種入力 ==> ()
    検索条件を入力する(a企業名, a企業コード, a業種) ==
      (
          is not yet specified;
      )
      post i検索条件入力 = mk_検索条件入力(a企業名,a企業コード,a業種);

    public 検索ボタンを押下する : () ==> ()
    検索ボタンを押下する() ==
      (
          検索処理(i検索条件入力);
      )
      post (len i企業一覧 > 0 and not i警告表示)
           or (len i企業一覧 = 0 and i警告表示);

-- ============================================
-- Actions
    public 検索処理 : 検索条件入力 ==> 検索結果
    検索処理(a検索条件) ==
      (
          is not yet specified;
      )
      pre not ( a検索条件 = mk_検索条件入力("","",<選択なし>) )
      post card(RESULT) >= 0;
```

- 具体的な処理は未定義

VDMの適用（2／3）

- 機能仕様の記述
  - 画面アクション明細をインプットに記述

```
-- ================================================
-- Events
      public 検索条件を入力する : 企業名入力 * 企業コード入力 * 業種入力 ==> ()
      検索条件を入力する(a企業名, a企業コード, a業種) ==
        (
          i検索条件入力 := mk_検索条件入力(a企業名,a企業コード,a業種);
        )
        post 検索条件が入力されている();

      public 検索ボタンを押下する : () ==> ()
      検索ボタンを押下する() ==
        (
          if 入力チェック() then (
              検索処理(i検索条件入力)
          );
          if card(i検索結果) > 0 then (
              i企業情報管理機能.i検索x一覧画面.初期表示(i検索結果);
          )else if card(i検索結果) = 0 then(
              ダイアログ表示("「企業情報は見つかりませんでした」");
          );
        )
        post 検索結果が1件以上である() or 検索結果が0件で警告を表示();
```

陽仕様

- 「検索ボタンを押下」した時の具体的な処理を記述

VDMの適用（3／3）

- 操作シナリオをテストケースとして記述

```
public scenario1 : () ==> ()
scenario1() == (
        def - = new IO().echo("*** -- 基本シナリオ ***\n") in skip;

        i検索x一覧画面.検索条件を入力する("oo株式会社","7070",<選択なし>);
        i検索x一覧画面.検索ボタンを押下する();
        i検索x一覧画面.検索結果から企業を選択する("7070");

        assertTrue(
            FSequence`Index[検索結果1件]
                (mk_検索結果1件("oo株式会社","7070"))(i検索x一覧画面.i企業一覧) > 0
        );
```

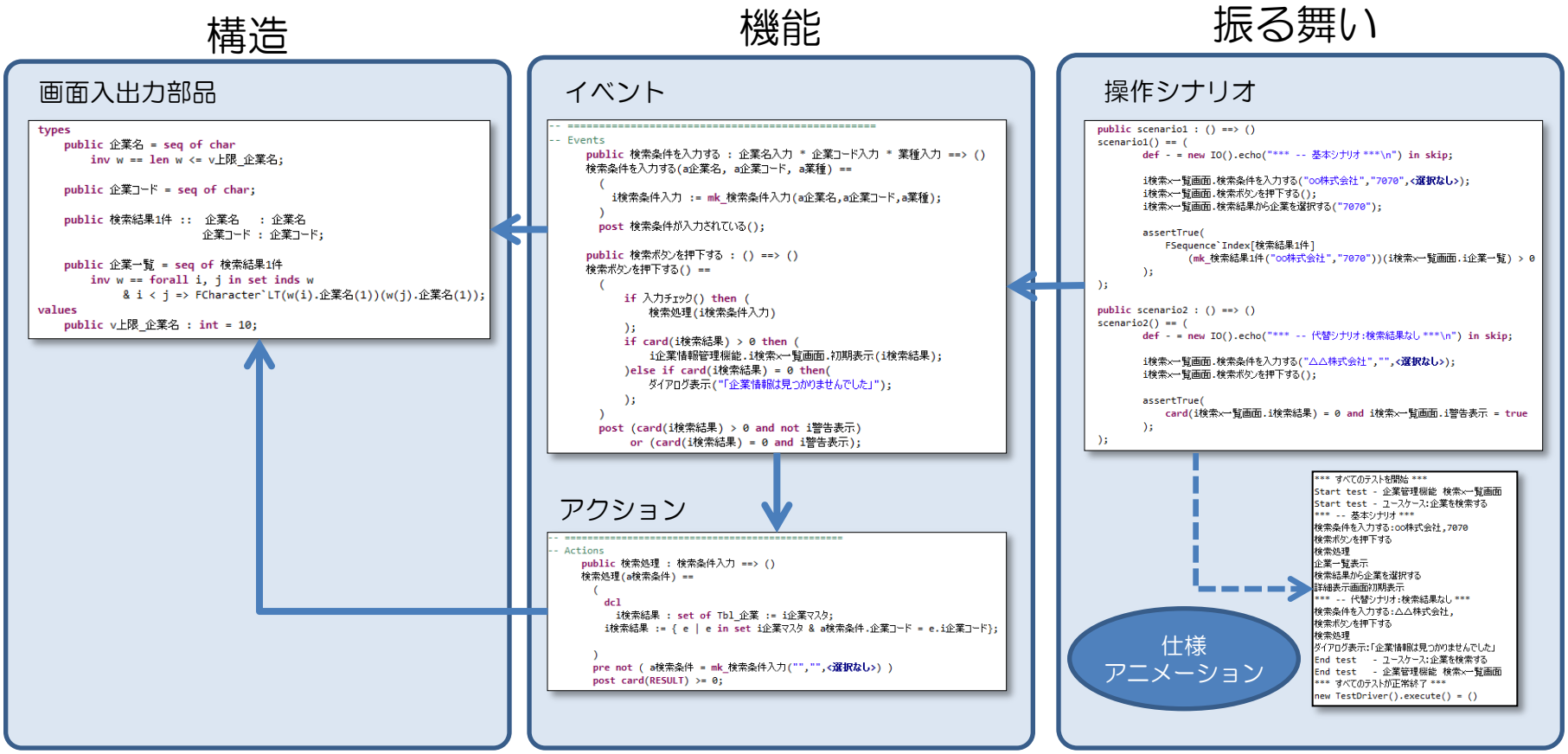
・定義したイベントを用いて、
ユーザー操作シナリオを記述

ツール上で
実行

```
Error Log | VDM Quick Interpreter | Console
<terminated> [Debug Console] New_configuration [VDM PP
*** すべてのテストを開始 ***
Start test - 企業管理機能 検索x一覧画面
Start test - ユースケース:企業を検索する
*** -- 基本シナリオ ***
検索条件を入力する:oo株式会社,7070
検索ボタンを押下する
検索処理
企業一覧表示
検索結果から企業を選択する
詳細表示画面初期表示
*** -- 代替シナリオ:検索結果なし ***
検索条件を入力する:△△株式会社,
検索ボタンを押下する
検索処理
ダイアログ表示:「企業情報は見つかりませんでした」
End test   - ユースケース:企業を検索する
End test   - 企業管理機能 検索x一覧画面
*** すべてのテストが正常終了 ***
new TestDriver().execute() = ()
```

結果

- 画面機能仕様書に含まれる「画面入出力項目一覧」、「画面アクション明細」、「画面レイアウト」に付記されている処理概要・操作手順、「共通ルール」をＶＤＭで記述・テストすることができた



仕様の曖昧さ・不足の例①

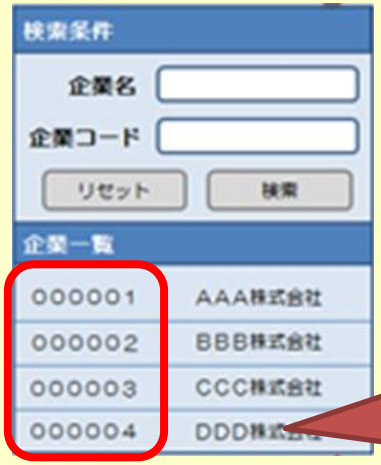
- **記述途中に発見**
  - 検索結果にどの項目を表示するか曖昧

画面入出力項目一覧

項目名	I/O	…	説明
企業一覧	O	…	検索結果を一覧で表示

検索結果の何を表示する？

画面レイアウト



企業コード？
その他のコード？
ただの連番？

定義

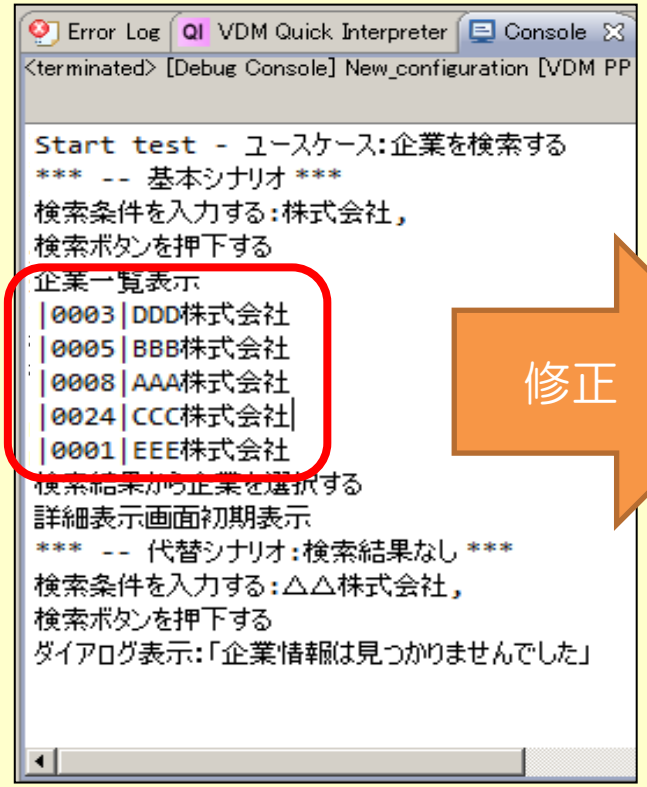
VDM記述

```
types
    public 企業名表示 = seq of char
        inv w == len w <= v上限_企業名表示;

    public 企業コード表示 = seq of char;

    public 検索結果1件 :: 企業名    : 企業名表示
                          企業コード : 企業コード表示;

    public 企業一覧 = seq of 検索結果1件

values
    public v上限_企業名表示 : int = 10;
    public v上限_企業一覧 : int = 5;
```

仕様の曖昧さ・不足の例②

- ツール実行、テスト時に発見
  - 企業一覧（検索結果）の並びや件数が不定

テスト結果



修正

VDM記述

```
public 企業一覧表示 : set of Tbl_企業 ==> ()
企業一覧表示(a検索結果) ==
  (
    def
        検索結果1件集合 =  {mk_検索結果1件(e.i企業名,e.i企業コード) | e in set a検索結果}
    in
        i企業一覧 := FSet`AsSequence[検索結果1件](検索結果1件集合);
  )
 post forall i, j in set inds i企業一覧
        & i < j => FCharacter`LT(i企業一覧(i).企業コード(1))(i企業一覧(j).企業コード(1))
      and len i企業一覧 <= v上限_企業一覧;
```

「企業コード順に並んでいること」
事後条件に追記・修正

まとめ

- 画面機能仕様書へＶＤＭを適用すると以下のような仕様の曖昧さや不足を排除する効果が期待できる
  - 記述する過程で、画面項目名が明確に定義され、思い込みや誤解の防止に効果がある
  - 想定するシナリオに沿って動かすことでき、誤りや曖昧と思われる個所に気付くことができる