資料1

# 疾患登録管理構築ガイドライン

# 別冊

厚生労働科学研究費補助金　難治性疾患等克服研究事業（難治性疾患等政策研究事業（難治性疾患政策研究事業））「今後の難病対策のあり方に関する研究」研究班

平成 26 年 12 月 20 日

目次

1. 概要

本書は、複雑な項目構造となる疾病定義をDBに登録し、自動的に以下のような機能を持たせることにより、疾病定義や疾病項目の追加・変更に伴うメンテナンスのコストを下げることを目標とする。

(1) 疾患データ入力フォームHTMLの雛形の自動生成

(2) 疾病定義IDに基づいた疾患データの保存

- 項目更新時の共通バリデーションの自動化

(3) 疾患データの取得

- フィルターIDを指定した項目のAND絞り込み

(4) 構造XMLの自動生成

## 2. 開発の流れ

疾患登録管理構築における開発の流れの概略図を下記へ示す。

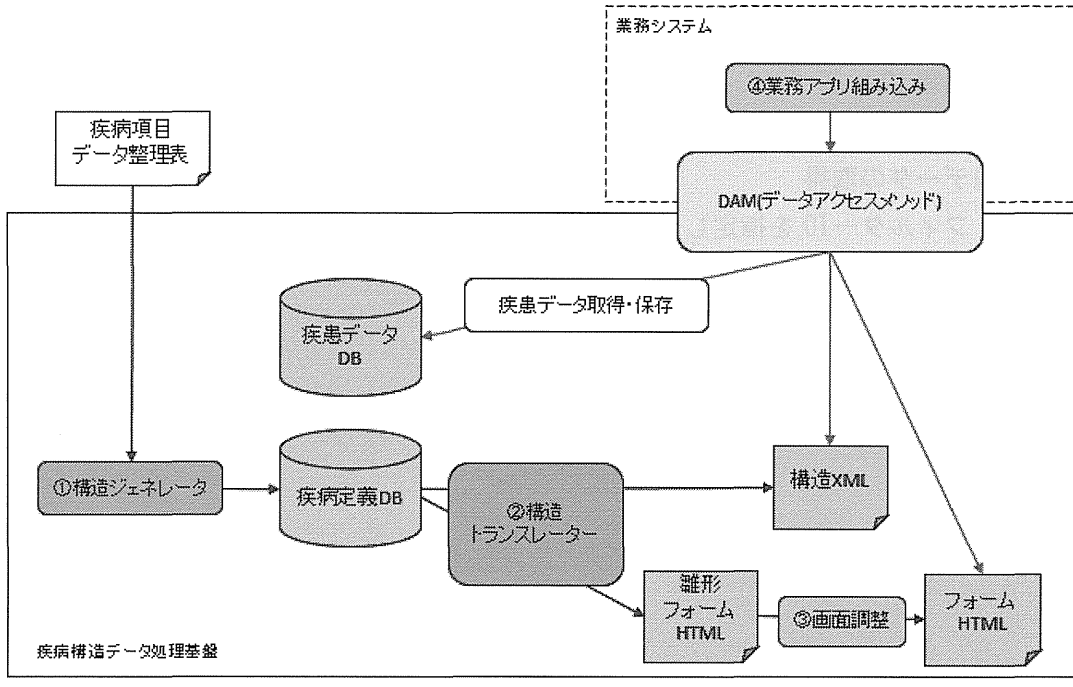


図 1：開発の流れ

(1) 構造ジェネレータを使用した疾病定義 DB の構築

(2) 構造トランスレーターを使用した疾病定義 DB から雛形フォーム HTML と構造 XML を生成

(3) 雛形フォーム HTML の修正

(4) 業務アプリに疾患データ保存・取得処理を組み込む

## 3. 疾病定義 DB の構築

### 3.1. 疾病定義 DB

疾病定義 DB は大きく以下の 2 つのテーブルで構成される。

- 疾病項目テーブル
- 疾病定義テーブル

表 1：疾患項目テーブル

| 項目名 | 説明 | 設定値 |
|---|---|---|
| 疾病項目 ID | 項目を表す ID | 任意 |
| データ型 | データの型を示す | 文字列<br>数値<br>年月日<br>年月<br>列挙型 |
| 次元性 | フィールドに対してデータが単数または複数 | 単数<br>複数 |
| バリデーションデータ | 入力可能な定義 | 桁数<br>範囲 |

表 2：疾病定義テーブル

| 項目名 | 説明 | 設定値 |
|---|---|---|
| 疾病定義 ID | 疾病を示す ID | 任意の文字列 |
| 疾病項目 ID | この疾病定義が保持する疾病項目 ID | 疾病項目テーブルへの参照 |
| 項目の階層パス | 項目の階層パスを / 区切り | 臨床試験/理学所見/身長 |
| 必須 | 必須項目を示す | 1->必須 |
| ISO 階層パス | ISO の階層パスを / 区切り | iso1/iso2/iso3 |

### 3.2. 共通バリデーション

疾病定義 DB により共通バリデーションは、データ格納時に自動的に行われる。
共通バリデーションは大きく分けて以下の 2 種類がある。

- 項目別バリデーション
- 疾病定義別バリデーション

#### (1) 項目別バリデーション

項目毎に固定となり、どの疾病でも同様に動作する。 以下バリデーションの種類を示す。

表 3：項目別バリデーション

| 種類 | 種類 | 説明 |
|---|---|---|
| 数値型 | 数値不正 | 正しい数値かチェック |
| | 範囲 | 数値型のみで数値の下限と上限を設定 |
| 文字列型 | 桁数 | 最大桁数を定義 |
| 年月日型 | 日付不正 | 日付が正しいがチェック |

#### (2) 疾病定義別バリデーション

同一の項目でも疾病定義毎に設定できる。 以下の種類がる。

表 4：疾病定義別バリデーション

| 種類 | 説明 |
|---|---|
| 必須 | データが正しく入力されている必要がある |

4.　疾病定義 DB からフォーム HTML の雛形と構造 XML を生成

4.1.　フォーム HTML の雛形

疾病定義 DB の内容から HTML フォームの雛形を生成する。

4.2.　構造 XML の生成

疾病定義 DB の内容から構造 XML を生成する。

5.　雛形フォーム HTML の修正

5.1.　フォーム HTML の雛形

フォーム HTML の雛形には以下の機能が組み込まれている。

- 項目の入力要素<input>タグなど（共通バリデーション定義込み）
- 構造の階層を反映したテーブルレイアウト構造
- タブ選択による画面切り替えと DRAFT データの更新処理

生成されるフォーム HTML の雛形は dkv-resource と呼ばれる JavaScript や css を参照し、動作する。

表 5：dkv-resource のファイル内容一覧

| ファイル | 説明 |
|---|---|
| dkvApp.js | クライアントサイド共通バリデーション処理<br>タブ押下での画面切り替えと保存処理<br>DkvServlet と通信するモジュール |
| form.css | フォーム雛形用のスタイルシート |
| jquery-1.11.1.js | JQuery を内部で使用<br>詳細は http://jquery.com/を参照 |
| jquery.inputmask.js | JQuery 入力文字規制プラグイン 取得先 |
| angular.js | AngularJS を内部で使用<br>詳細は https://angularjs.org/を参照 |

雛形のままのフォーム HTML で項目のデータ保存・取得が可能な状態ではあるが、生成された雛形に対して手動で以下のような調整を施す必要がある。

(1) レイアウトの調整

階層構造そのままでテーブルがネストした形式で生成される。場合に応じレイアウトを調整する必要がある。

(2) 表示・非表示ロジックの付与

特定の項目値のみ入力を促したい場合に入力フォームを表示する。

(3) 自動計算ロジック

入力値がある特定の値の場合、他の入力値を補助的に設定する。

Note

現状ではフォーム HTML の雛形は毎回生成されるので、調整後のフォーム HTML に手動でマージしていく必要がある。

将来的には、カテゴリ毎に調整後のテンプレート化した HTML を保存しておくことにより、疾病構造 DB の変更に対して動的に HTML 生成する予定。

5.2. 表示・非表示ロジックの付与

表示ロジック対象と表示ロジック値を踏まえた、表示ロジックの対象となるデータ項目には、他のデータ項目との関連性を考慮する必要がある。また、共通バリデーション処理の対象外のデータ項目がある。それら入力画面上での表示/非表示の動作制御が必要となるデータ項目は、HTML テンプレートへの修正が必要となる。

HTML テンプレートへの修正は、AngularJS の仕様に基づき、HTML タグに ng-if 属性を付加して、以下の例のように対応することが可能である。

タグを利用した、{項目統合 ID}を持つデータ項目が、選択肢に「1」をとる場合にのみ、{表示内容}の入力項目を表示

```
<span ng-if="data.{項目統合 ID}==1">{表示内容}</span>
```

例 1：HTML テンプレートへの修正

5.3. 自動計算ロジックの付与

ある項目の値が変更された時に、別の項目の値が自動的に設定されるようにするには angularjs の ng-change 属性を使用する。

変更を検知する ng-model 属性が付与されたタグに ng-change 属性を付与する。

aaa 項目の値が 37 以上なら bbb 項目の値が 1 それ以外なら 2 を設定する。

```
<input ng-model="data.aaa" ng-change="data.bbb = (data.aaa>=37 ? 1 : 2)">
```

例 2：自動計算ロジックの付与

6.　業務アプリからの疾患データ取得・保存

6.1.　本登録テーブルとドラフトテーブル

疾患データのテーブルは疾患データ ID がキーの「本登録テーブル」とユーザ ID がキーの「DRAFT テーブル」の 2 種類が存在する。

本登録テーブルのデータは直接編集できない。チェックアウト->モディファイ(DRAFT)->コミット方式で本登録テーブルを更新する。

本登録テーブルとドラフトテーブルはレコードの内容は同じで、以下の共通項目を保持する。

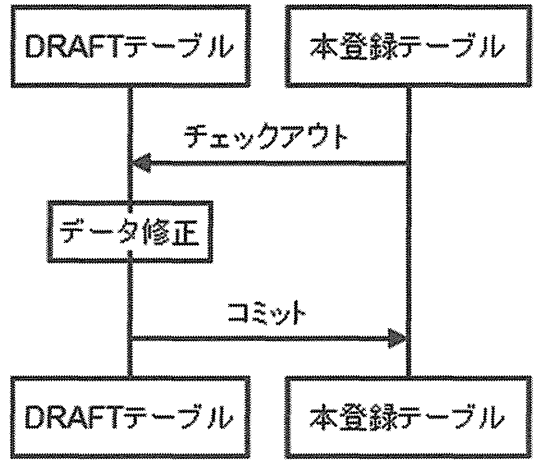


図 2：本登録テーブルとドラフトテーブル

表 6：本登録テーブルとドラフトテーブルの共通項目

| 項目名 | DKV 項目名 | 説明 |
|---|---|---|
| 疾病 ID | フォーム ID (formID) | 更新時のフォーム ID |
| 患者データ ID | レコード ID (recordID) | レコードを識別する ID |
| 疾患データ ID 履歴 | レコード ID 履歴 | 複製元のレコード ID を直近順のマルチバリューで保持 |

API はダイナミックキーバリューデータアクセスメソッド(DkvDAM)という汎用的な DB サービスを使用して構築される。

6.2. DkvDAM.java

DkvDAM を Java でラッピングした I/F。 詳細は JavaDoc を参照のこと。

6.3. DkvServlet

業務アプリは DkvDAM を直接呼び出すことができる。また J2EE にて DkvServlet を使用しブラウザ ⇔ 業務アプリ ⇔ 難病 API の通信部分を気にすることなくフォーム編集機能を実装することが出来る。下記に実装手順を示す。

6.3.1. 実装手順

(1) jp.go.nisp.dkv.jar を WebApp のライブラリとして組み込む。

(2) dkv-resource フォルダを WebContent 直下に配置する。
フォームが利用する JavaScript ライブラリと css が含まれる。

(3) AbstractDkvServlet を継承した DkvServlet を作成。

```
package aaa.bbb;
import jp.go.nisp.servlet.AbstractDkvServlet;
public class DkvServlet extends AbstractDkvServlet {
    @Override
    protected String getDraftId(HttpServletRequest request) {
        // セッション等からユーザを識別する ID を返す
        return "userid";
    }
    @Override
    protected boolean canAccess(HttpServletRequest request, String recordId) {
        // 指定のレコード ID はユーザがアクセス可能か
        return true;
    }
    @Override
    protected boolean afterCommit(String recordId, String customData) {
        // レコード ID が返ってくるので業務アプリ側の DB と同期する処理を実装する
        // customData は文字列以外の場合は JSON にパースされている
        return true;
    }
}
```

(4) web.xml に DkvServlet を登録。

```
<servlet>
   <servlet-name>Dkv</servlet-name>
   <servlet-class>aaa.bbb.DkvServlet</servlet-class>
</servlet>
<servlet-mapping>
   <servlet-name>Dkv</servlet-name>
   <url-pattern>/dkv/*</url-pattern>
</servlet-mapping>
```

(5) 業務アプリ画面の HTML 作成。

① 画面 HTML が表示される前に目的に合わせて以下の DkvDAM の API を使用して下書きデータを用意する。

表 7：業務アプリ画面の HTML 作成

| 目的 | 呼び出す DAM |
|---|---|
| 新規疾患データ作成 | DkvDAM#newDraft(ドラフト ID,疾病定義 ID) |
| 既存疾患データの編集 | DkvDAM#checkout(ドラフト ID,疾患データ ID) |
| 既存疾患データを元に新規作成 | DkvDAM#checkout(ドラフト ID,疾患データ ID,新しい疾病定義 ID) |

② 入力フォームを表示したい箇所に iframe を書く。

```
<html>
<body>
  <iframe      src="/gyomuapp/dkv/getform?formId=060"      target="diseaseFrame"
scrolling="no" onload="afterLoadForm()">
</body>
</html>
```

③ 業務アプリ HTML にコミットボタンを配置しイベントにコミット処理起動を書く。

```
window.diseaseFrame.dkvCommit(customData);
```

6.3.2. フォーム HTML の javascriptAPI 仕様

内部の通信は非同期通信となる。非同期通信の結果は戻り値として[JQuery Deferred Object]で返される。

成功・失敗のハンドリング方法は JQuery Deferred Object API マニュアルを参考のこと。

コミット dkvCommit(validation,customData)

表 8：引数

| 変数 | 説明 |
|---|---|
| validation | true<br>正当性チェックを行いクリアしないと本登録テーブルに保存されない<br>false<br>正当性チェックを行わずに本登録テーブルに保存される |
| customData | DkvServlet#afterCommit()内で受け取る<br>文字列以外の場合は JSON にエンコードされる |
| 戻り値：JQuery Deferred Object | |

6.3.3. DkvServlet を使用した処理別の流れ

(1) 一時保存データ取得

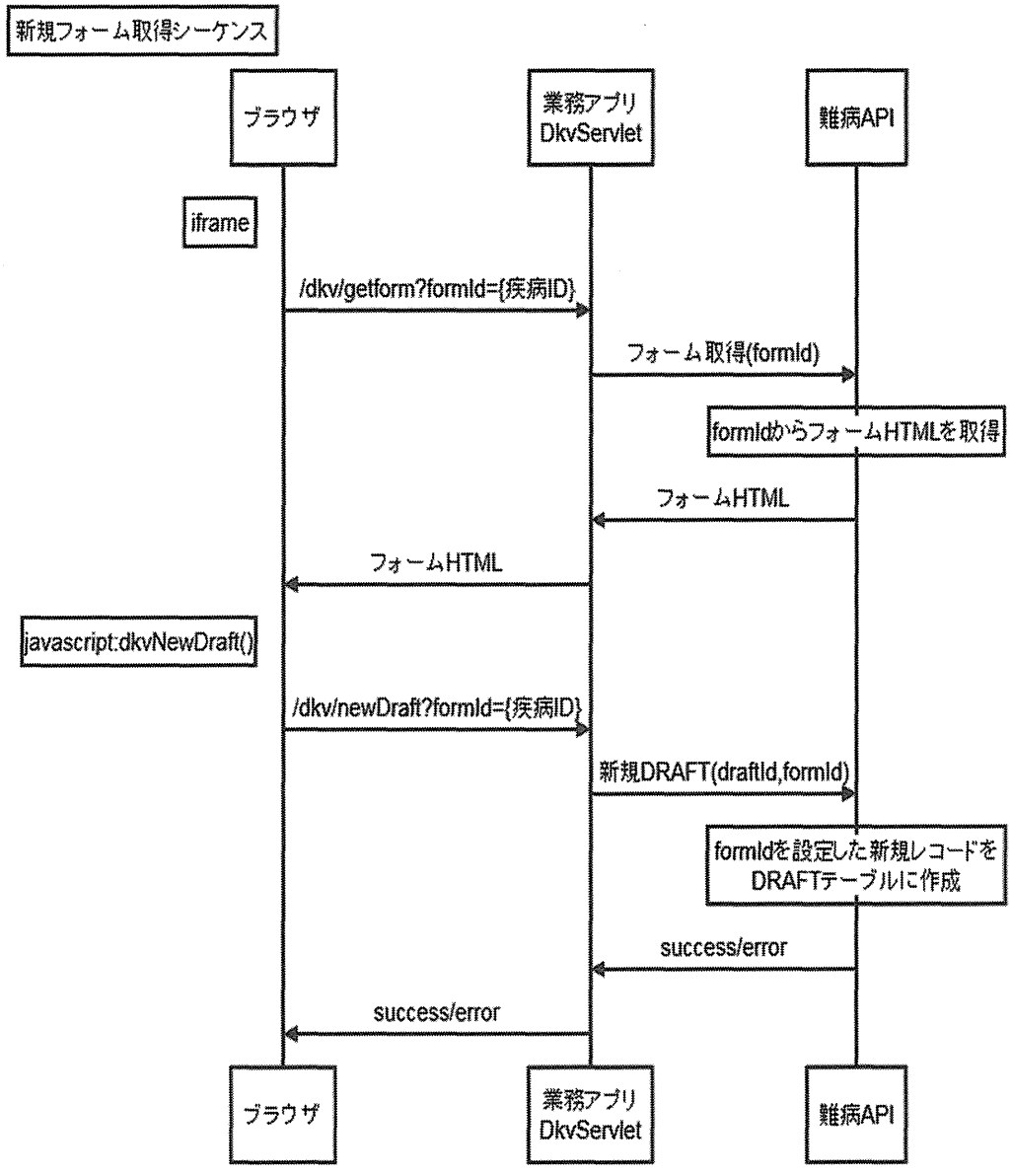


図 3：一時保存データ取得

(2) 一時保存データ修正

タブのフォーカスが変わると自動的に以下のシーケンスが実行される。

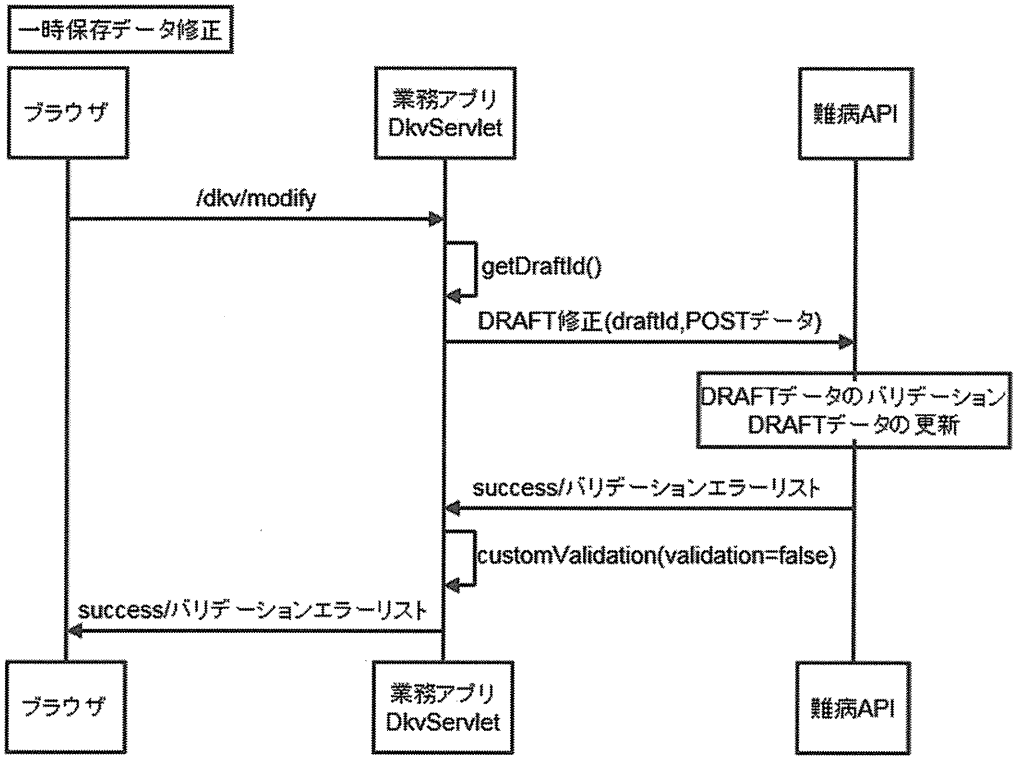


図 4：一時保存データ修正

(3) データ保存（バリデーションチェックあり）

dkvCommit(validation=true)で実行した場合に以下のシーケンスが実行される。

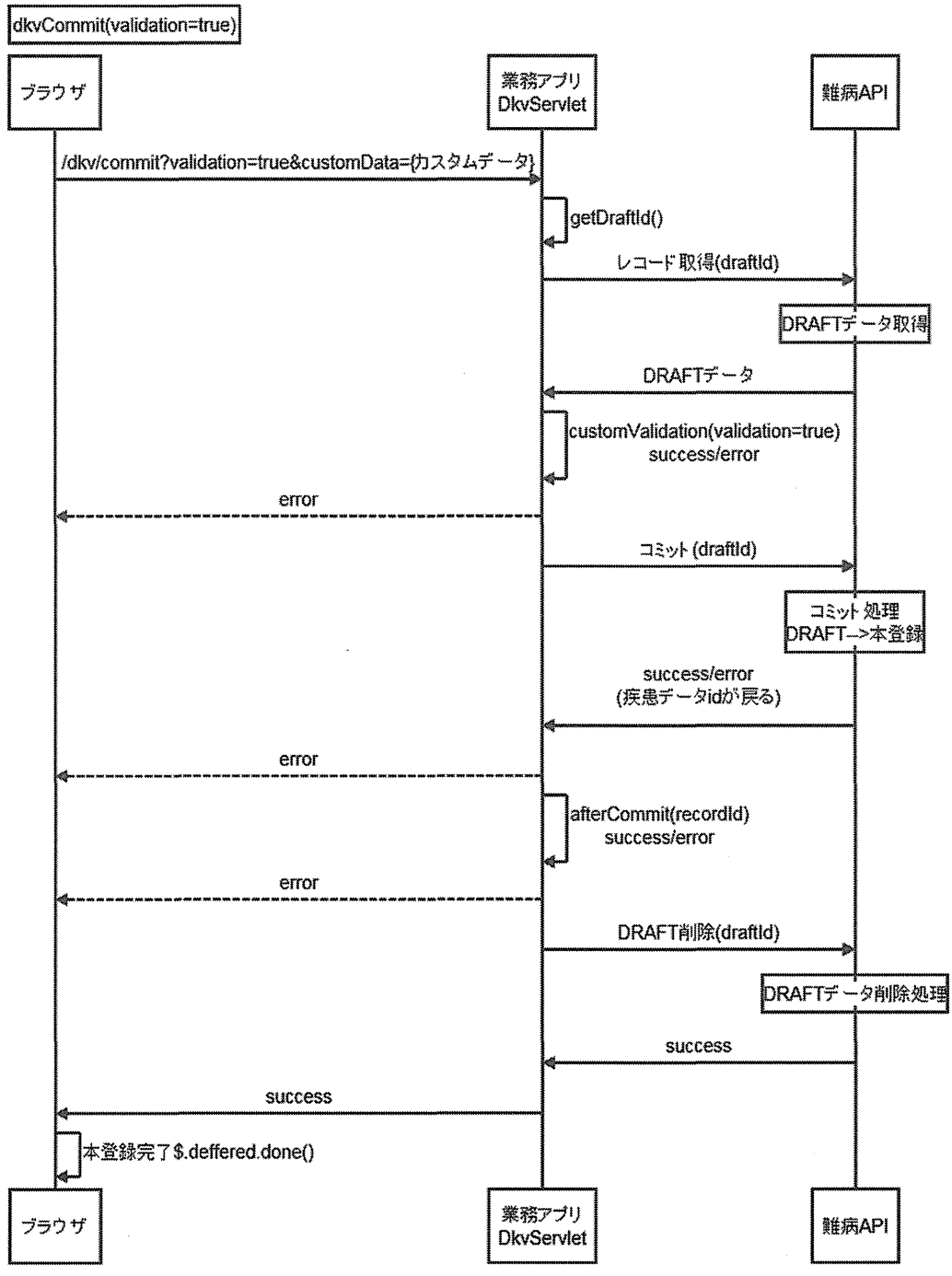


図 5：データ保存（バリデーションチェックあり）

(4) データ保存（バリデーションチェックなし）

dkvCommit(validation=false)で実行した場合に以下のシーケンスが実行される。

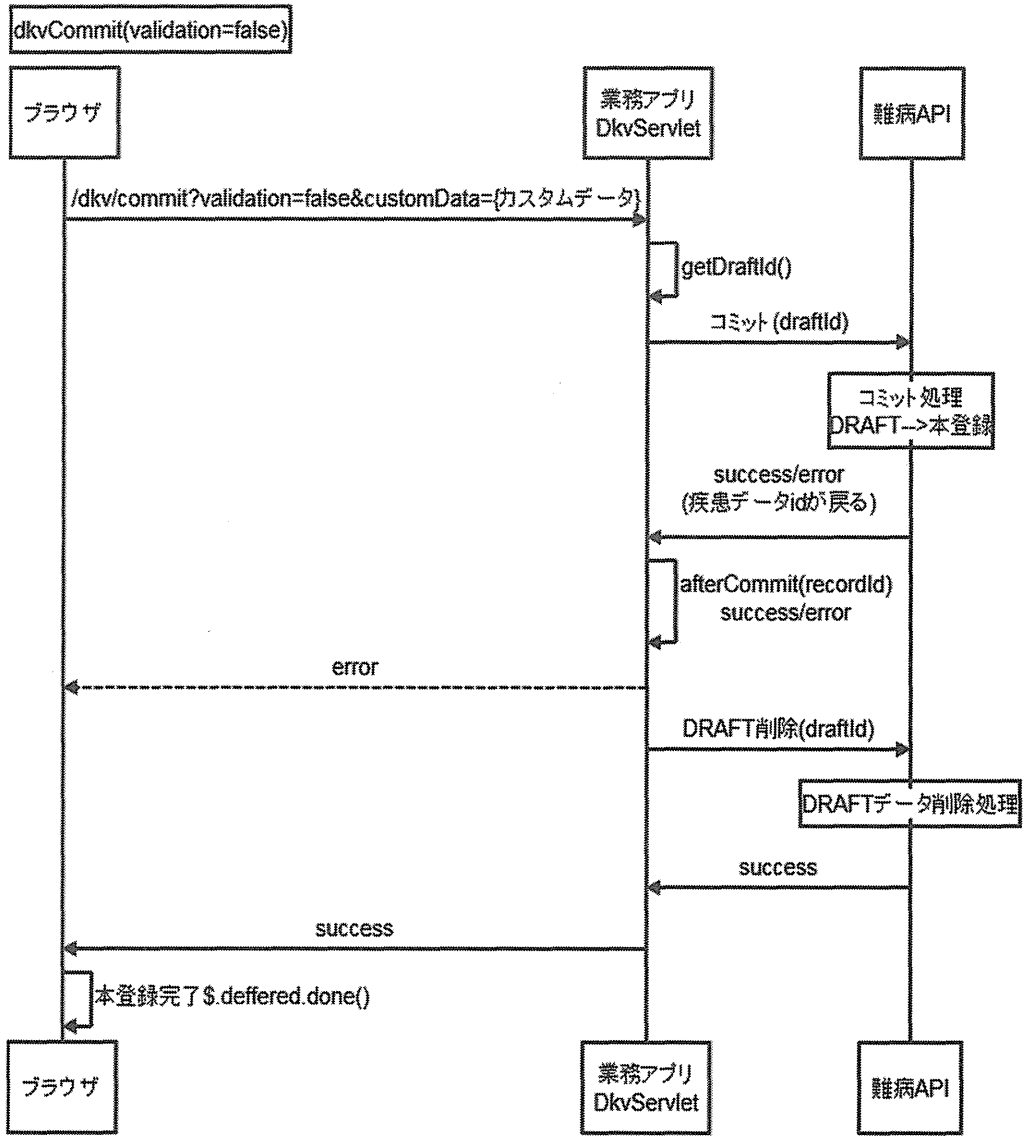


図 6：データ保存（バリデーションチェックなし）

## 6.4. DkvDAM サブルーチン仕様

DkvDAM は OpenQM のサブルーチンにて実装される。
QMClient API の RPC 機能を利用してサブルーチンを呼び出すことにより使用する。
QMClient API は C 言語のライブラリである。よって、そのライブラリを呼び出すことができる言語から呼び出し可能である。
詳細は http://www.openqm.org/docs/ の QMClient API を確認のこと。

Java 以外で実装する場合や、通信仕様の詳細を確認する場合は以下の DAM サブルーチン仕様を確認すること。

### (1) フォーム取得

指定のフォーム ID に対応するフォーム HTML を取得する。

表 9：フォーム HTML の取得

サブルーチン名:DKV.GET.FORM

| 引数番号 | IN/OUT | 名前 | 説明 |
|---|---|---|---|
| 1 | IN | フォーム ID | フォーム ID |
| 2 | OUT | 結果 | 0->成功<br>1->フォーム ID が存在しない<br>2->その他エラー |
| 3 | OUT | フォーム HTML | HTML(utf-8) |

(2) 新規ドラフトレコード作成

ドラフトテーブルに新しいレコードを追加する。 ドラフト ID が存在する場合は上書きする。

表 10：新規ドラフトレコード作成

サブルーチン名: DKV.NEW.DRAFT

| 引数番号 | IN/OUT | 名前 | 説明 |
|---|---|---|---|
| 1 | IN | ドラフト ID | 新規生成するドラフトテーブルのキー |
| 2 | IN | フォーム ID | 新規レコードのフォーム ID |
| 3 | OUT | 結果 | 0->成功<br>1->その他エラー |

(3) チェックアウト

本登録テーブルのレコードからドラフトテーブルにコピーする。
ドラフト ID が存在する場合は上書きする。
新しいフォーム ID が指定された場合は、コピーしたレコードに対して以下の操作を行う。

- レコード ID を空にする
- レコード ID 履歴に元レコード ID を追加
- フォーム ID に新しいフォーム ID を設定

表 11：チェックアウト

サブルーチン名: DKV.CHECKOUT

| 引数番号 | IN/OUT | 名前 | 説明 |
|---|---|---|---|
| 1 | IN | ドラフト ID | コピー先のドラフトテーブルのキー |
| 2 | IN | レコード ID | コピー元の本登録テーブルのキー |
| 3 | IN | 新しいフォーム ID | コピーする時に新しいフォーム ID を設定 |
| 4 | OUT | 結果 | 0->成功<br>1->キーが存在しない<br>2->その他エラー |

(4) ドラフトレコード修正

ドラフトテーブルの指定レコードの指定項目の値を変更する。
バリデーションエラーが存在してもドラフトテーブルに値は格納される。

表 12：ドラフトレコード修正

サブルーチン名: DKV.MODIFY.DRAFT

| 引数番号 | IN/OUT | 名前 | 説明 |
|---|---|---|---|
| 1 | IN | ドラフト ID | ドラフトテーブルのキー |
| 2 | IN | 更新対象項目リスト | @FM 区切り |
| 3 | IN | 値リスト | @FM 区切り |
| 4 | OUT | 結果 | 0->成功<br>1->キーが存在しない<br>2->バリデーションエラー |
| 5 | OUT | バリデーションエラー | 引数 4 が 2 の場合に[バリデーションエラー]<br>リスト@FM 区切り |

表 13：バリデーションエラー

| V-no | 項目名 | S/M | 説明 |
|---|---|---|---|
| 1 | 項目 ID | S | 項目を識別する値 |
| 2 | エラータイプ | M | required->必須エラー<br>maxlength->最大桁数エラー<br>minlength->最小桁数エラー<br>range->数値範囲エラー<br>date->不正な日付エラー<br>number->不正な数値エラー<br>custom->その他エラー(オプションで:の後にメッセージが付けられる) |