6B-5

スマートデバイス用DB連携アプリケーション向けフレームワークの提案

石倉 直弥† 伊藤 俊明† 菊地 大介†
†株式会社 日立ソリューションズ東日本

1. はじめに

近年，スマートフォンやタブレット型端末(スマートデバイス)の業務活用ニーズが高まっている．スマートデバイスの業務活用の例として既存業務データベース(DB)との連携が挙げられる．既存システムの多くはDBを持ち，DBへの参照や編集を行っている．これらの操作をスマートデバイスから行えると業務データを社外でも活用できる．

スマートデバイスの業務活用に際して，初期導入時の開発コストが大きいという問題がある．既存の業務システムとの連携機能やスマートデバイス用アプリケーションの新規開発が必要になるためである．

2. DB連携アプリケーション向けフレームワーク

上記の課題解決のために，既存業務DBにアクセスするスマートデバイス向けフレームワークを提案する．本フレームワークはDBへの参照や編集操作をスマートデバイスから行うための機能を提供する．構成を図1に示す．本フレームワークにより定義ファイルの作成だけで新規アプリケーションの開発が可能となり，開発コストの削減ができる．

2.1. 機能概要

(1) データ通信機能

データ通信機能はサーバDBとスマートデバイス間でデータを送受する機能である．テーブル構成などのデータ送受のために必要な情報は，データ定義ファイルに記述されており，それらに基づいて，テーブルやレコードの生成，参照，更新，削除を行う．

(2) 画面生成機能

画面生成機能は，テーブルデータから画面を自動生成する機能である．画面に表示するデータ項目や画面間の遷移関係を画面定義ファイルで決定する．

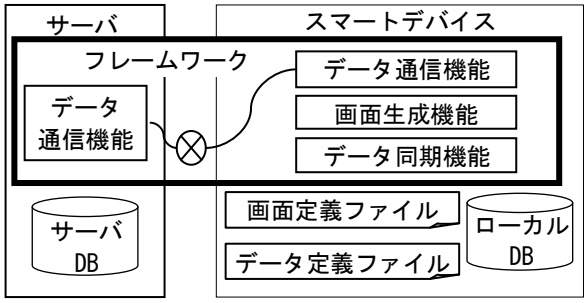

図 1 DB連携アプリケーション向けフレームワーク

Database Access Framework of Smart Device Application
Naoya Ishikura†, Toshiaki Ito†, Daisuke Kikuchi†
†Hitachi Solutions East Japan, Ltd.

生成される画面はテーブルの一覧参照を行う一覧画面，テーブル内のレコードについて参照と編集を行う詳細画面の2種類がある．

(3) データ同期機能

データ同期機能は，ローカルDBとサーバDBを同期する機能であり，電波状況が悪いなどの理由でネットワークへ接続できない場合を想定した機能である．同期するテーブルはデータ定義ファイルによって任意に指定できる．

2.2. 定義ファイル概要

(1)データ定義ファイル

データ定義ファイルはサーバDBが持つテーブルとそのテーブル構成および同期するテーブルについてXMLで記述する．

データ定義ファイルの例を図2に示す．<DataTable>はDBが保持している一つのテーブルを表す．<Column>はテーブルの列情報(ID,データ型など)を表す．<DataTable>と<Column>はサーバDBに応じて複数定義できる．また，<AutoSync>で同期するテーブルを設定する．その際，<UploadTables>と<DownloadTables>によって，アップロードおよびダウンロードするテーブルを個別に指定できる．図2の例ではアップロードとダウンロードを同じ「工事」というidのテーブルに設定している．

```
<DataDefinition>
  <DataTables>
    <DataTable  id="工事">
      <Columns>
        <Column id="工事ID"    type="String"/>
        <Column id="工事名"    type="String"/>
        <Column id="開始予定日"  type="Date"/>
        <Column id="終了予定日"  type="Date"/>
      </Columns>
    </DataTable>
    <!— 省略 -->
  </DataTables>
  <AutoSync>
    <UploadTables>
      <SyncTable id="工事"/>
    </UploadTables>
    <DownloadTables>
      <SyncTable id="工事"/>
    </DownloadTables>
  </AutoSync>
</DataDefinition>
```

図 2 データ定義ファイルの例

```
<ScreenDefinition>
  <Screens>
    <Screen id="screen1" tableId="工事">
      <ListView>
        <ListViewItem colId="工事 ID"/>
        <ListViewItem colId="工事名" />
        <OnClickTransit>
          <DestinationScreen type="list"
            screenId="screen2"/>
          <DestinationScreen type="detail"/>
        </OnClickTransit>
      </ListView>
      <DetailView>
        <DetailViewItem colId="工事 ID "/>
        <DetailViewItem colId="工事名"/>
        <DetailViewItem colId="開始予定日"/>
        <DetailViewItem colId="終了予定日"/>
      </DetailView>
    </Screen>
    <Screen id="screen2" tableId="工程">
      <!-- 省略 -->
    </Screen>
  </Screens>
</ScreenDefinition>
```

図 3 画面定義ファイルの例

(2)画面定義ファイル

画面定義ファイルではテーブル情報の内，画面に表示するデータや，画面間の遷移情報について記述する．図 3 は画面定義ファイルの例である．

<ListView>は一覧画面を表す．<ListViewItem>は表示する各データ項目となる．この例では，工事 ID と工事名を列として持つ一覧が作成される．また，一覧から特定のレコードを選択した場合，画面遷移が発生する．遷移先は<DestinationScreen>によって設定し，複数設定可能である．8 から 9 行目の<DestinationScreen>は別に定義する screen2 の一覧画面への遷移を，10 行目は同一画面(screen1)の詳細画面（DetailView）への遷移となる．

<DetailView>では詳細画面を定義する．また，<DetailViewItem>は表示するデータ要素を表す．この例では工事テーブルの工事 ID,工事名，開始予定日，終了予定日を詳細画面に表示する．

2.3. 画面表示例

本フレームワークを用いた画面の例を図 4 に示す．図 4(a)は図 3 の<ListView>から作成された一覧画面である．表示された一覧から任意のレコードをタップすることで(b)の遷移先選択画面が表示される．ここで工事(詳細画面)を選択すると(c)のように図 3 の<DetailView>で定義した詳細画面が表示される．表示されるユーザインターフェースはデータ定義ファイルの<Column>で指定されている type 属性を基に，編集操作に適したものが自動的に生成される．

(a)一覧画面　(b)遷移先選択　(c)詳細画面

図 4 画面例

表 1 Java による開発ステップ数

機能	想定ステップ数
画面	3,000 ステップ/15 画面
通信機能	2,000 ステップ
データ同期機能	1,000 ステップ
計	6,000 ステップ

表 2 定義ファイル作成のステップ数

機能	想定ステップ数
データ定義ファイル	75 ステップ/5 テーブル
画面定義ファイル	250 ステップ/5 テーブル
計	325 ステップ

3. 評価

本フレームワークと Java それぞれを使ったアプリケーション開発の工数・期間を比較する．

開発は Android SDK[1]を使用して行う．参照するデータテーブル数を 5，各テーブルが持つ列数を 10 とする．必要な画面数は，テーブル毎の一覧画面と詳細画面（計 10 画面），その他の管理画面（5 画面）の計 15 画面とする．

Java を用いた開発工数の見積り(表 1)では新規開発時のステップ数は約 6,000 ステップとなり，生産性を当社標準値である 2.5 キロステップ/人月とすると約 2.4 人月の開発期間が必要となる．

一方，本フレームワークを用いた場合，開発者が行う作業は各定義ファイルの作成だけとなる．定義ファイルの記述量(表 2)から工数は 3 人日となる．検証結果より，本フレームワークを用いることで，開発工数を 2.4 人月（約 48 人日）から 3 人日と，およそ 1/16 に大きく短縮できる．

4. おわりに

本稿では既存業務 DB にアクセスするスマートデバイス向けフレームワークを提案した．Java を使用する場合と比較し開発コストを大幅に削減できることを示した．

参考文献

[1]Android SDK | Android Developers, http://developer.android.com/sdk/index.html, Accessed 2012/12