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

Proposal of Framework for Smart Device Application to Cooperate with DB

ビジネス分野でのスマートデバイスの活用ニーズが増えつつある。しかしこれらは従来の PC などとは異なる特徴を持つデバイスであり，業務への適用に際し新規アプリケーションの開発や，既存システムとの連携機能の開発が必要となる。開発に要する初期コストが，業務へのこれらデバイス導入の障壁となっている。

本稿ではスマートデバイスの業務適用を容易にする技術の確立を目的に，これらのデバイスから任意の業務データベース(DB)の参照・編集を可能とする DB 連携アプリケーション向けフレームワークを提案する。本フレームワークを用いてアプリケーションを構築することにより，開発工数・期間を削減できる。

伊藤　俊明　Ito Toshiaki
菊地　大介　Kikuchi Daisuke

１．はじめに

海外および国内でのスマートデバイスの普及が拡大している。ここでのスマートデバイスとは，スマートフォンやタブレット端末など，タッチパネルを備えた板状のモバイル端末を指す。その利便性や汎用性から，業務でこれらを利用したいと言うニーズが増えている。スマートフォンおよびタブレット端末の 2011 年度の法人向け国内出荷台数は約 180 万台で前年比約 200%増， 2012 年度は 320 万台の出荷が予想されている [1)]。

従来，業務に活用するモバイル端末としてはノート PC やハンディターミナルがあった。スマートデバイスはこれらノート PC ともハンディターミナルとも異なる特徴を持つ。ノート PC と比較した場合，軽量で長時間のバッテリー駆動が可能など屋外での利用に適するが，一般にキーボード，マウスなどハードウェアによる入力支援手段を持たないことから多量の文字入力や細かな画面操作は困難であり，PC と異なるユーザインタフェース(UI)などの検討が必要となる。

このようにスマートデバイスは従来にない特徴を持っており，業務適用に際して新たなアプリケーションの開発が必要となる。またすでに多くの顧客企業では業務を支援する IT システムを活用しており，業務に適用するには既存の IT 資産とこれらデバイスの連携が必要となる。

本稿では，業務へのスマートデバイスの適用を容易にする技術の確立を目的に，これらのデバイス上で動作する DB 連携アプリケーション向けフレームワークを提案する。

２．業務へのスマートデバイス適用の課題

多くの業務アプリケーションは，DB に格納されたデータを参照，加工し，再び DB に格納するといった処理を行っている。スマートデバイスの業務での主な活用方法としては，既存の業務システムで利用している DB のデータを，これらを用いて社外からでも参照／編集可能とすることが挙げられる。

スマートデバイスの業務適用に際しては，上記を実現する画面，通信機能を持った業務アプリケーションの開発が必要となる。しかし，アプリケーションの新規開発には多くのコスト，期間を要する。これが初期導入コストの増大，ひいては業務適用の障壁となっている。

多様な顧客へスマートデバイスを適用し，ソリューションとして展開するために，アプリケーションを低コスト・短期間で構築できる技術が求められている。

３．DB 連携アプリケーション向けフレームワーク

低コスト,短期間でアプリケーションを構築するには,開発・テスト期間の削減が重要である。そのためには開発者がソースコードをできるだけ記述せずにアプリケーションを構築できるフレームワークが必要である。

そこで，本稿ではデータ通信，画面生成，データ同期の機能モジュールと，これら機能の動作を決定する外部定義ファイルからなるアプリケーションのフレームワークを提案する。

本フレームワークにより，定義ファイルの変更のみでアプリケーション開発ができるようになり，低コスト・短期間での開発が可能となる。

3.1　フレームワークの提案

DB と連携する業務アプリケーションに求められるデータ通信，画面生成，データ同期の基本機能をフレームワークとして提供する。また，本フレームワークでは,これらの機能の動作をデータ定義ファイルと画面定義ファイルで決定する。フレームワークのモデルを図 1 に示す。

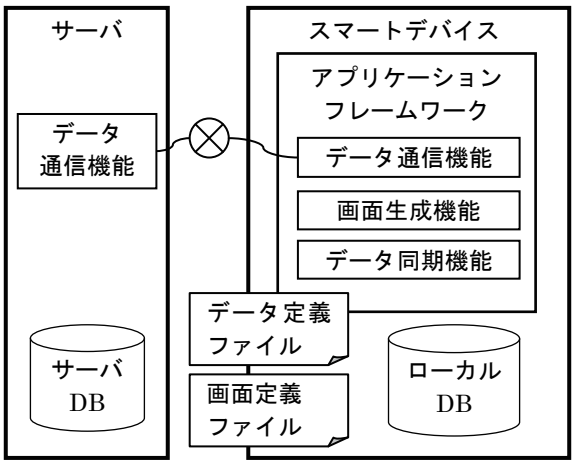

図 1　DB 連携アプリケーション向けフレームワーク

(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授受するための機能である。授受するデータの内容は，外部の定義ファイルであるデータ定義ファイルにより決定する。DB のテーブル構成に基づいて定義ファイルで指定されたデータテーブルと,生成,読み取り,更新，削除などデータの操作種別に基づき，REST (Representational State Transfer)[2] 形式のリクエスト／レスポンスを生成し,データ通信を行う。これにより,DB のテーブル構成に合わせたデータ通信機能の構築が容易となり,アプリケーションの開発工数を削減できる。

(2) 画面生成機能

画面生成機能は，スマートデバイス上でテーブルデータを表示する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外部の定義ファイル（画面定義ファイル）で決定する。

スマートデバイスで業務データの参照・編集を行うには，業務データのテーブル構成に対応し，かつこれらのデバイスに適した UI が必要である。タッチパネルでは PC のような細かな画面操作は難しいことから ，1 画面に複数のデータリストや操作用コントロールが含まれる複雑な画面構成を避け，本フレームワークではテーブルの一覧参照や絞り込みを行う「データ一覧画面」,個別データの参照編集を行う「データ詳細画面」からアプリケーションを構成する。

画面定義ファイルで上記画面の表示項目，画面間の遷移関係を指定することで，表示および遷移の制御の手続きを記述することなく容易に画面を構築できるため，画面の開発工数を削減できる。

(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る。電波の状況が悪い屋外作業などでのアプリケーションの利用を想定した機能である。同期するデータテーブルはデータ定義ファイルにより指定できる。クライアントアプリケーションはデータ定義ファイルに基づきスマートデバイス内のローカル DB にサーバ DB のデータをダウンロードし，オフラインでのデータの参照／編集を可能とする。これによりこれらデバイスがネットワークに接続できない状況でも業務データの参照や記録が可能となる。

(4) 定義ファイル

本フレームワークが使用する定義ファイルはデータ定義ファイルと画面定義ファイルの 2 種類である。

データ定義ファイルでは，クライアントアプリケーションが通信や同期の対象とするデータ項目を，DB のテーブル構成に基づきテーブル ID やカラム ID を用いて定義する。画面定義ファイルではアプリケーションの画面に表示するデータ項目や表示順序，画面間の遷移関係をテーブル ID, カラム ID を用いて定義する。データ通信,画面生成，データ同期の各機能は，これらの定義ファイ

ルに基づいて動作する。また，定義ファイルの記述は，XML 形式を採用した。これにより開発者の閲覧・編集を容易にする。

このように，サーバ DB と連携するスマートデバイス向けアプリケーションで求められる基本機能をフレームワークとして提供することで，定義ファイルを変更するだけのカスタマイズでアプリケーション開発ができるようになり，低コスト・短期間での開発を実現する。

3.2 フレームワークの適用例

提案したフレームワークを用いて構築したアプリケーションの例を示す。

(1) 開発・動作環境

表 1 に開発・動作環境を示す。スマートデバイスには Android OS[3]が動作するデバイスを選択した。Android は国内のスマートフォン契約台数で 5 割超のシェアを持っており [4]，業務用のスマートデバイスとして今後利用される可能性が高い。一方，サーバ側には業務で広く利用されている Windows 環境を使用した。開発言語は，スマートデバイス側は Java，サーバ側は C#を用いた。

表 1 試作システムの開発・動作環境

	スマートデバイス	サーバ
OS	Android OS (Ver2.3)	Windows OS
開発言語・フレームワーク	Java/ Android SDK[5]	C#/ ASP.NET MVC

(2) 定義ファイルの記述例

クライアントアプリケーションの動作を定めるデータ定義ファイル，画面定義ファイルの記述例をそれぞれ図 2，図 3 に示す。

図 2 はデータ定義ファイルの例である。<DataTable>要素はアプリケーションが扱う一つのデータテーブルを表している。<DataTable>の下位にある<Row>要素はテーブル内のカラム（ID，データ型など）を表す。<DataTable>要素および<Row>要素は業務で扱うデータテーブルに応じ複数定義可能である。図 2 の例は，工事 ID，工事名，開始予定日などをカラムとして持つ，工事テーブルの定義を表している。

図 3 は画面定義ファイルの例である。<ListView>要素は一覧画面に表示するデータ項目を，<DetailView>は詳細画面で表示するデータ要素を表している。また，<OnClickTransit >要素は，一覧画面で画面がクリック（画面遷移の操作）された場合の遷移先画面のタイプや ID を指定している。遷移先は複数定義可能である。また遷移に伴う絞り込み条件も定義できる。

3 行目から 11 行目までの ListView の定義は，工事テーブルの工事 ID，工事名をリストに表示し，リストがクリックされた場合の遷移先が screen2, および detail（詳細画面）であることを定義している。12 行目から 17 行目までの DetailView の定義は，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ことを定義している。

```
<DataTables>
  <DataTable   id="工事" >
    <Rows>
      <Row   id="工事コード"   type="String" />
      <Row   id="工事名" type="String" />
      <Row   id="開始予定日"   type="Date" />
      <Row   id="終了予定日"   type="Date" />
        :
    </Rows>
  </DataTable>
</DataTables>
```

図 2 データ定義ファイルの例

```
<Screens>
  <Screen   id="screen1"   tableId="工事" >
    <ListView>
      <ListViewItem rowId="工事 ID" />
      <ListViewItem rowId="工事名" />
      <OnClickTransit>
        <DestinationScreen   type="list"
          screenId="screen2" />
        <DestinationScreen   type="detail" />
      </OnClickTransit>
    </ListView>

    <DetailView>
      <DetailViewItem   rowId="工事 ID " />
      <DetailViewItem   rowId="工事名" />
      <DetailViewItem   rowId="開始予定日" />
      <DetailViewItem   rowId="終了予定日" />
    </DetailView>
  </Screen>

  <Screen   id="screen2"   tableId="工程" >
    :
  </Screen>
</Screens>
```

図 3 画面定義ファイルの例

(3) 定義ファイルに基づく画面例

図 2，図 3 の定義ファイルに基づいて本フレームワークが構築したクライアントアプリケーションの主要画面の例を示す。

(a) データ一覧画面

データ一覧画面では DB の任意のテーブルのデータをリスト形式で表示する。本フレームワークでは項目を縦に並べるリスト表示形式を採用しており，リスト全体が画面に入りきらない場合は縦にスクロールして残項目を表示する。リストの一要素内に表示する項目は，画面定義ファイルにより指定できる。一要素内に複数の項目を表示することも可能である。

図 4 は，図 3 の画面定義ファイルに基づき生成された一覧画面の例である。画面定義ファイルの ListView の定義に基づき，工事テーブルの工事 ID，工事名をリストに表示している。

図 4　データ一覧画面の例

図 5　画面遷移の例

図 5 はデータ一覧画面から他画面への遷移のイメージを表している。一覧リストの要素（行）をクリックすることで，クリックされたレコードのデータ詳細画面，またはクリックされたレコードに関連する他テーブルのデータ一覧画面に遷移する。クリック時の遷移先は画面定義ファイルで指定可能であり，遷移先として複数の候補を記述することも可能である。遷移先が複数ある場合は遷移先を選択するダイアログを表示し，ユーザに遷移先の選択を促す。

図 5 は，図 3 の画面定義ファイルの OnClickTransit の定義に基づく画面遷移の例である。図 5 左側の画面では工程テーブルの一覧画面，または工事テーブルの詳細画面への遷移の選択を示すダイアログを表示している。ダイアログの項目を選択することで図 5 右側の工程テーブルの一覧画面，または工事テーブルの詳細画面へ遷移を行う。

(b) データ詳細画面

データ詳細画面ではデータの 1 レコードの表示と編集が可能である。データ詳細画面に表示する項目や表示順は，画面定義ファイルで指定できる。

図 6 は，図 3 の画面定義ファイルの DetailView の定義に基づく詳細画面の例である。定義に基づき工事テーブルの工事 ID，工事名，開始予定日，終了予定日を表示しており，編集が可能である。また，データの編集時にはデータ定義ファイルで指定されたデータ型に対応する入力用インタフェースを自動的に表示する。この例では，図 2 のデータ定義ファイルで String 型を指定した工事名に対して図 6 右上の画面のようにソフトウェアキーボードを，Date 型を指定した開始予定日に対しては図 6 の右下の画面のように日付入力用のデイトピッカーを表示している。

図 6　データ詳細画面の例

このように，提案するフレームワークでは画面の表示項目や画面間の遷移が定義ファイルにより定義可能であり，顧客の業務データに応じた画面がコーディングレスで構築できる。また画面に表示されるデータのサーバからの取得，ローカル DB への保存処理なども，定義ファイルの記述に従い構築される。

4．評価

提案するフレームワークを用いたアプリケーション開発によって，同様のアプリケーションの新規開発と比較して，低コスト・短期間で開発できることを検証した。

4.1 前提条件

屋外での建築物などの検査業務にスマートデバイスを適用するケースを想定する。開発する画面，機能を以下のように定める。

スマートデバイスから参照するデータテーブル数は 5 テーブル（案件情報，建築物の情報，顧客情報，検査項目，担当者マスタなど），1 テーブルが持つカラム数は 10 カラムとする。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），およびデータの同期などのための管理画面（5 画面）の，計 15 画面とする。社内 DB との連携のためのサーバとの通信機能，電波状況が悪い建築現場での作業を考慮しローカル DB へのデータ保存・同期機能も開発対象とする。

4.2 開発工数・期間の比較

提案するフレームワークを用いたときの開発工数・期間と，新規開発時の開発工数・期間を比較した。

新規開発を行う場合には，画面数に応じた開発工数，および通信機能，同期機能の開発工数が必要となる。

Android アプリケーションの画面および通信機能の開発に必要なソースコードのステップ数を，これまでの試作実績に基づき表 2 のように定めた。画面の新規開発には，1 画面平均 200 ステップ， 15 画面では 3,000 ステップを要する。スマートデバイスとサーバ間の通信機能の新規開発には 2,000 ステップ，デバイス内のローカル DB のアクセスおよび同期機能の開発には, 1,000 ステップを要する。

以上より新規開発時のステップ数は約 6,000 ステップとなる。Android アプリケーションの主要な開発言語は Java であり，生産性を約 2.5 キロステップ/人月と仮定すると，アプリケーションの新規開発には約 2.4 人月が必要となる。開発担当者 2 名で，主要な機能単位である画面と通信・同期機能に分割して開発を実施した場合，1.2 ヵ月程度の開発期間が見込まれる。

表 2　新規開発ステップ数

機能	想定ステップ数
画面	3,000 ステップ/ 15 画面 （平均 200 ステップ/1 画面）
通信機能	2,000 ステップ
ローカル DB・同期機能	1,000 ステップ
計	6,000 ステップ

一方，本試作システムで 4.1 の想定業務のアプリケーションを構築する場合，開発者が行う作業はデータ定義ファイル，画面遷移定義ファイルの作成となる。

定義ファイルの想定記述量を表 3 に示す。データ定義ファイルにはデータテーブルおよびデータカラムの記述を行い, 5 テーブル全体で 75 要素(75 ステップ)程度となる。

画面定義ファイルではテーブルの一覧画面と詳細画面毎に表示する項目，および画面間の遷移の記述を行う。試作実績より，一覧画面で 20 ステップ，詳細画面で 30 ステップ程度, 5 テーブルの一覧・詳細画面の合計で 250 ステップ程度となる。データ定義および画面定義の合算では 325 ステップ程度である。

表 3　定義ファイル作成の想定ステップ数

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 （平均 15 ステップ/1 テーブル）
画面定義ファイル	250 ステップ/5 テーブル （ 20 ステップ/一覧画面, 30 ステップ/詳細画面）
計	325 ステップ

試作実績よりこの規模の XML 定義ファイルの作成，およびテストに要する工数は 2 から 3 人日程度である。また小規模作業のため SE 一人で上記作業を実施することを想定し必要期間は 3 日程度である。

検証結果より，提案するフレームワークを用いた開発と新規開発を比較した場合，工数では 2.4 人月を 3 日程度に，開発期間は 1.2 ヶ月程度を 3 日程度に大幅に短縮できる可能性があることが分かった。

５．おわりに

今後もスマートデバイスの普及および業務利用は拡大していくと予想される。これらを業務に円滑に適用するための技術や製品，ノウハウを持つことは IT ベンダにとっての強みの一つとなり得る。

これらデバイスの業務適用を容易にするため，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を提案した。本フレームワークは，これらデバイスの DB 連携アプリケーションで求められる基本機能を提供する。各機能の動作は定義ファイルによって決定される。このようにすることで，定義ファイルを作成するだけで容易にアプリケーションを構築できる。

また，本フレームワークを適用することで新規開発に比べ開発工数や期間を大幅に削減できることを示した。

今後，定義ファイルの作成支援機能，対応 OS の拡充などのエンハンス，および実システムでの適用評価を通し，スマートデバイス分野で強みとなる自社製品と事業の創出につなげる。

参考文献

1) 2012 法人向けスマートデバイス関連ビジネスの全貌，株式会社富士キメラ総研, 2011

2) Leonard Richardson，他 ： RESTful Web サービス，オライリー・ジャパン, 2007

3) Android，http://www.android.com/ （Accessed 2012/09）

4) スマートフォン市場規模の推移・予測，http://www.m2ri.jp/newsreleases/main.php?id=010120120313500 （Accessed 2012/09）

5) Android SDK， http://developer.android.com/intl/ja/sdk/index.html （Accessed 2012/09）

伊藤　俊明　　1993 年入社
研究開発部
スマートデバイス活用アプリケーションの研究・開発
ito@hitachi-to.co.jp

菊地　大介　　2009 年入社
研究開発部
農作物の生育管理，スマートデバイス活用アプリケーションの研究・開発
daisuke.kikuchi.01@hitachi-to.co.jp