Androidアプリによる食品管理システムの構築と適用評価

Android application development for an unmanned food store and its trial release

飯山　大樹†　　中所武司†
Hiroki Iiyama　　Takeshi Chusho

1. はじめに

近年，スマートフォンやタブレット端末の市場は著しい成長を続けている．これらのデバイスには数年前には想像できないような機能が備わっている．

その機能を実現する上で欠かせないのが「アプリ」である．その開発技術は，従来のＰＣ用のアプリケーション開発技術と共通する部分もあるが，デバイスに依存する部分も多い．そこで，我々は，Android アプリとして食品管理システムを構築するとともに，その運用に関して問題点，課題点を解決するという開発経験を通じて，スマートフォンやタブレット端末に特有の開発技法について検討する．

2. 研究目的とアプローチ

2.1 研究目的

スマートフォンアプリの開発と運用保守に関する技術的課題を考察するために，例題アプリとして，研究室で利用する食品管理システムを取り上げ，ケーススタディを行う．

2.2 例題アプリケーション

例題アプリケーションとして取り上げた食品管理システムは，われわれの研究室で使用するものである．

現状では，買い出し係が飲み物やカップ麺，飲料水などの食品をスーパーでまとめて購入し，冷蔵庫などに保管している．ゼミ生は必要に応じてこれらの食品を消費することができるが，それを所定の用紙に記録する．会計係は，その記録に基づいて，時々，ゼミ生から集金する．このように，現在，食品購入の記録を紙媒体で行っているため，いろいろと手作業が必要であり，ミスやトラブルも発生しやすくなっている．

そこで，今回，タブレット端末を利用した食品管理システムを開発して，これらの問題を解決するとともに，その開発経験を通じて，アプリの開発技法を研究することにした．

なお，以下のような理由で，OS としては Android を選択した．

- 開発環境が自由であること．
- 開発し実際に利用するまでが容易であること．

3. 食品管理システムの基本機能の実装

3.1 システム概要

食品管理システムのユースケース図を図 1 に示す．アクターとしては，すでに説明した学生，買い出し係，会計係のほかに，利用する学生を管理するユーザ管理者を新たに導入した．これらのアクターが利用できる機能は以下のようになる．

学生は，以下の，食品を購入するための基本的な機能を利用できる．

① 食品購入
② 食品購入の取り消し
③ 個人履歴の閲覧
④ 在庫情報閲覧

会計係は，学生から請求金額を徴収するため，以下の機能を利用できる．学生の精算が終了したら，RESET ボタンを押すことにより，今まで蓄積してきた未精算の合計金額を 0 に戻すことができる．

⑤ 支払い金額一覧の閲覧
⑥ 学生履歴の閲覧

ユーザ管理者は，学生を管理するために，以下の基本的な機能を利用できる．

⑦ ユーザ登録
⑧ ユーザ削除

買い出し係は，扱う食品の仕入れと在庫管理を担当するため，以下の機能を利用できる．なお，食品の登録には，JAN コードを入力することでデータベースに存在するか否かを確認できる重複登録防止機能がある．JAN コードとは，バーコードの 13 桁あるいは 8 桁の数字のことであり，食品を一意に識別できる．

④ 在庫情報閲覧
⑨ 食品情報の編集
⑩ 食品の登録

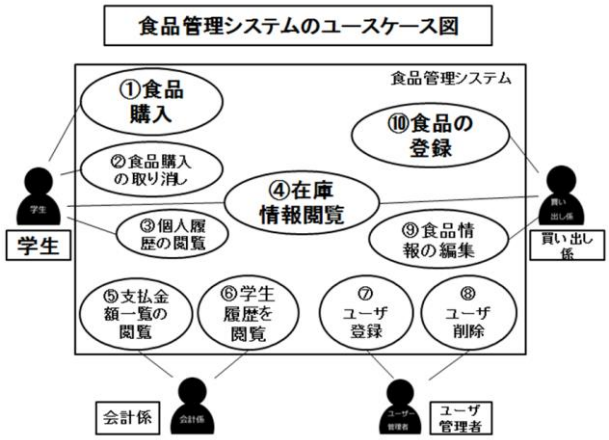

図 1：基本機能を実装したシステムのユースケース図

3.2 データベース概要

データベースには SQLite を利用している．SQlite は，MySQL や PostgleSQL と同じデータベース管理システムであり，アプリケーションに組み込んで利用される軽量のデータベースである．構築する食品管理システムは，複数のユーザが一つのタブレット端末を利用する形なので，

†明治大学大学院理工学研究科基礎理工学専攻情報科学系　ソフトウェア工学研究室

ローカルな環境で動作する SQLite は今回のケースで適しているといえる．

今回の食品管理システムの構築に関して，作成したテーブルは以下の 3 つである．

(a) Customer テーブル

学生の情報に関するテーブルである．学生 ID を primary key とし，password，氏名，役職，合計支払金額の 5 つのカラムを持つ．

(b) Item テーブル

食品に関するテーブルである．商品を一意に識別できる JAN コードを primary key とし，商品名，売価，在庫数の 4 つのカラムを持つ．

(c) Orderhistory テーブル

食品購入に関するテーブルである．購入した時刻を注文 ID とし，primary key としている．学生 ID，JAN コード，購入個数，購入金額の 5 つのカラムを持つ．

3.3 ID・パスワード関連の安全対策

今回のアプリケーションは，一台のタブレット端末にインストールして，ゼミ室に設置するので，端末の操作は研究室に属する学生のみである．そこで，ID・パスワードを複雑で安全性の高いものにする必要ないと考え，ID・パスワード共に 4 桁の数字とした．ID は，前半 2 桁が自身の研究室の期（現 4 年は 19 期生），後半が名前順の順番を ID にした（例：19 期生 1 番飯山→1901）．パスワードは任意の 4 桁とする．

しかし，ID・パスワードに関して，図 2 に示すような問題点があった．すなわち，ID・パスワードを入力し，ログインした後，目的の機能を利用し，最後にログアウトを行った後，バックボタンを繰り返し押すと，入力した ID・パスワードが入力フォームに表示されたままの ID・パスワード入力画面が表示される．つまり，第 3 者に ID・パスワードが知られてしまう可能性があることが判明した．

この問題点の解決策として，端末のバックボタンを本来のバックボタンとして機能させない方法が考えられる．端末のバックボタンを押すと前の画面に戻るように設定されているが，これをシステム終了の確認ダイアログを表示するように変更した．変更後の仕組みを図 3 に示す．

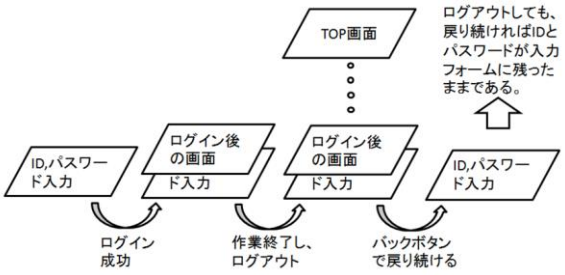

図 2：ID・パスワードの問題点

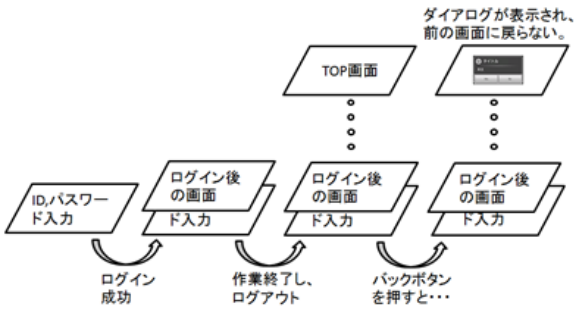

図 3：ID・パスワードの問題点の解決策

図 3 のように変更することで，ID，パスワードの流出を防ぐことが可能になった．

4. ユーザの試用評価

基本機能を実装した段階で利用するユーザ(25 名)を対象にアンケートを行った．設定した質問に対して 5 段階の評価での回答を求めた．結果を表 1 に示す．

基本機能を実装した段階であり，システム自体はシンプルな画面遷移なので，ユーザにとっては理解しやすかったと思われる．そのため，操作の習得のしやすさや操作のしやすさに関する項目では 4 以上の評価が得られた．一方，画面遷移の効率に関する項目の評価が低かった．アンケート結果や，試用した感想などから得られたシステムの問題点としては，

- 稀に強制終了がある．
- 前画面に戻る方法がわかりにくい．
- 画面遷移が遅い．

などが指摘され，優先すべき改善点が明確になった．

表 1：アンケート結果

タッチパネルは操作しやすいですか.	4.28
画面遷移は軽快ですか.	3.78
文字入力は操作しやすいですか.	4.07
ゼミ室に設置したら，１人で問題なく作業できると思いますか.	4.28
慣れてしまえば，サクサク操作できるシステムだと思いますか.	4.42
１年後でも，説明なしに１人で操作できるシステムだと思いますか.	4.14
このシステムには満足ですか.	4.14

5. ユーザ評価に基づく改良版の実装

5.1 アンケート結果に関する問題点の解決策

(1) メモリオーバーによる強制終了について

システムを利用している最中にアプリが強制終了することがあった．画像がリスト形式で表示される画面でエラーが起こるので，画像の表示に関してエラーが起こっていると考えられたため，詳しく調べることにした．

画像の縮小率や，アプリが処理する画像の枚数などを変更し，アプリが強制終了するかどうかを調査した．結果，ある一定の枚数以下の画像と，ある一定の縮小率以上にすることにより，強制終了がなくなった．この調査から強制終了する規則性はわからないが，原因が画像を扱うことによる OutOfMemoryError であることを判断した．アンケート調査中に何回か発生した強制終了は，画像を読み込む際に元の画質のまま読み込むことが原因であった．Android アプリにはメモリ制限があり，端末によって決まっている．元の画像の画質のまま画像を読み込むと OutOfMomoryError のエラー発生が高くなる．今回の強制終了に対する対策を，図 4 に示す．

以下に図 4 の流れに沿って説明する．

① 画像情報（サイズや容量など）のみを読み込む．画像自体を読み込むわけではない．
② 情報を元に画像の縮小率を求める．
③ 決定した縮小率を元に，元の画像を加工してから読み込む．
④ 利用しやすいようにさらに加工し，画像を利用する．

縮小率の求め方は以下のように設定した．

縮小率＝max(元画像の縦幅／表示する枠の縦幅，元画像の横幅／表示する枠の横幅)

この縮小率を求めることにより，実際にアプリで利用したい大きさに加工することができ，強制終了が発生しなくなった．

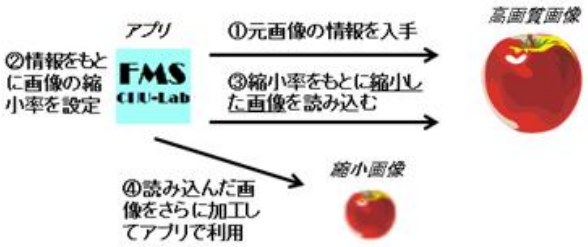

図 4：画像の読み込み方法

(2) 前画面に戻る方法がわかりにくい問題

アンケート結果から，各画面に「戻る」ボタンがあったほうがよいという感想が多くみられた．よって，すべての画面に「戻る」ボタンを設置した．

表示されている画面の終了に finish メソッドを利用する方法があるが，前の画面に戻るとデータが反映されていないなどの不都合を防ぐため，すべてインテント（新しい画面を生成すること）して前の画面に戻っている．

(3) 画面遷移の遅さについて

画面遷移が軽快ではないという意見が多かった．原因としては画像の読み込みによるものと考えられるが，OutOfMemoryError 対策としてそれぞれの画像に対して図 4 のような処理を行うので必然的に画面遷移の速度はより遅くなってしまう．

対策として，画面遷移処理中はダイアログを表示し，画像読み込みの処理中であることをユーザに通知している．

5.2 追加機能の実装

基本機能の他に新たな機能の追加を試みた．追加機能を実装する上で，新たに 2 つのテーブルを追加した．

(1) ユーザ情報変更機能

パスワードと名前を自身で変更できる機能である．パスワードに関しては今までのパスワードと新しいパスワードを入力する必要がある．

(2) ログイン・ログアウト記録機能

ユーザのログイン・ログアウトを記録しておく機能である．ユーザは意識することはない機能であり，記録しておくことで他ユーザの不正使用を防止する．本機能を実現するために，新たに Log テーブルを作成した．

(3) 食品評価機能

食品に対して利用者が簡易的に評価できる機能である．商品に対して GOOD あるいは BAD の評価を行うことができる．評価した履歴は閲覧することができ，アプリの TOP 画面で「最近評価された 5 つの食品」，「全ユーザの評価 BEST3」を閲覧することができる．本機能実現のために，新たに Rating テーブルを作成した．

(4) 合計金額の個別リセット

基本機能として全ユーザの一括リセットがあったが，個別にリセットできたほうが良いという意見があり，個別にリセットできるようにした．ユーザのリストをタッチすると合計金額をリセットできる．

(5) カメラ機能の追加

商品情報として画像データを表示するようにした．実現させるために，アプリからカメラを起動して撮影できるようにした．カメラ機能を実現する方法として，

- 構築したアプリ自体にカメラ操作機能を直接実装する方法
- 暗黙的インテントにより外部のカメラアプリを利用する方法

がある．メリット，デメリットは表 2 に示す．

表 2：カメラ機能実現方法の比較評価

	メリット	デメリット
アプリに直接実装	構築しているアプリに適したカメラを自由に作成できる．	自身でカメラ自体を作成するため，手間と時間がかかる．
外部のカメラアプリを利用	数行のコードで，強力な外部のカメラアプリを利用できる．	外部のカメラアプリは端末によって種類が変わるので，端末によっては不具合が起こる可能性がある．

最初に内部でカメラを実装することを試みたが，実装が難しく，かつ時間がかかるので，今回は外部のカメラアプリを利用する方法を試みた．仕組みを図 5 に示す．

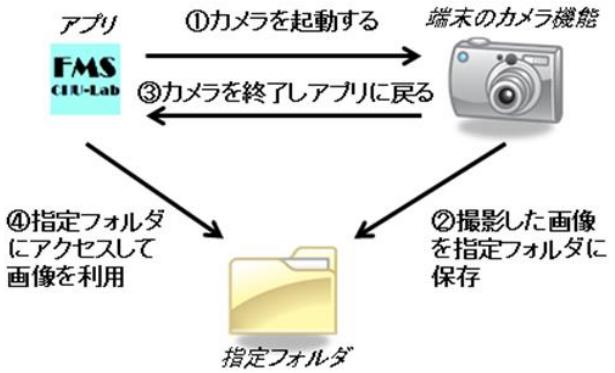

図 5： 外部カメラアプリ利用の仕組み

この方法により，外部カメラを利用して指定フォルダに撮影した写真を保存し，実際に写真をアプリで利用することができた．実際に写真を利用している画面の例を図 6 に示す．図 6 は購入可能な食品の一覧を示している．左端に食品の写真を掲載し，中央に食品名，右端に売価や在庫数などを表示している．購入した食品情報をタッチすると，購入個数入力画面に画面遷移し，購入個数入力後，食品が購入可能となる．

図 6： 写真利用の画面例

(6) Android アプリのメモリ解放

Android アプリは，モバイルに特化した Dalvik 仮想マシンを利用している．Dalvik 仮想マシンではメモリに関するエラーが発生する可能性がある．

Android アプリを起動すると多数のオブジェクトが生成される．アプリを終了すると，使用されていないオブジェクトは解放してくれるが，参照が残っているオブジェクトは解放されていないケースが存在する．改めてアプリを起動すると，解放されていないオブジェクトは再利用されず，新たなオブジェクトが生成される．これを繰り返すことで，OutOfMemoryError が突然発生する可能性がある．

対策として，アクティビティのライフサイクルのメソッドである onStop メソッドが呼ばれた際に，解放されにくいオブジェクトである List や Adapter や Listenner に関して，明示的に参照を切っている．画面遷移をすると onStop メソッドが呼ばれるため，onStop メソッドをオーバーライドして参照を切っている．以下が参照を切る例である．

```
@Override
protected void onStop() {
super.onStop();
back_button.setOnClickListener(null);
listView.setOnItemClickListener(null);
listView.setAdapter(adapter);
adapter = null;
if (waitDialog != null)
waitDialog.dismiss();
}
```

(7) ユーザビリティの向上

ユーザビリティとは，「使いやすさ」を意味する．学習のしやすさ，効率性，記憶しやすさ，エラー，主観的満足度の 5 つのユーザビリティ特性からなる[5].

画面構成をシンプルにし，補助文を多く付け加え，重要な文章に色をつけるなどにより，快適でわかりやすいシステム構築を心掛けた．

6. 考察

Android アプリとして簡易な食品管理システムを実装することは難しいことではない．基本的には画面遷移の方法と SQLite の利用方法さえ理解してしまえば実装は簡単である．しかし，運用することを前提として考えると，システムの構築に関して様々なことを考慮しなければならない．バリデーションやユーザビリティの向上，パスワードの流出を防ぐなどである．また，一定の期間システムを利用してもらうことで発生するエラーも Android アプリでは存在する．このエラーはデバッグでは発見しにくいエラーである．このような点が単にシステムを構築する場合と，ある程度運用まで視野にいれたシステムの構築の違いであると考える．

本研究では，当初「システム利用にストレスを感じないユーザインターフェース，及びシンプルな構造の達成」を掲げており，基本的な機能のみを実装することを考えていた．また，Android アプリを作成するうえで，どの程度の時間を要するのか不安であったからでもある．上記でも挙げたが基本的なシステムの構築は，実際には思ったより容易であった．

「追加機能を実装する上で他の機能を妨げることはない」ことに注目して，ユーザビリティ特性の 1 つである「主観的満足度：ユーザが個人的に満足できるよう，また好きになるよう，楽しく利用できなければならない」に注目し，食品評価機能を追加した．評価機能により，ユーザが注目する食品がわかるので，購入の際や買い出しの際に参考になり，楽しく利用できることを期待する．

画像の処理に関して，利用者数の多い有名なアプリなどは，画像の扱いの際に時間を取られることはあまりない．画像処理をバックグラウンドで処理させていると考えられる．本研究では行っていないが，よりよい解決策があると考えられる．

7. おわりに

本研究では，Android アプリによるゼミ室設置に特化した食品管理システムを構築した．アンケートを行い，問題点の改善を行った．また運用する上で新しく発生する問題点についての解決も行った．ユーザビリティ向上のため，基本機能の他に，様々な追加機能を実装した．

なお，今回開発した Android アプリの実装結果として，クラス数：46，XML ファイル数：37，表示画面数：34 である．

適用評価に利用したタブレット端末の仕様は，プラットフォーム：Android4.0，CPU：NVIDIA® Tegra® 3 モバイルプロセッサ，メモリ：1GB である．

参考文献

[1] 石山　陽士：「タッチパネルを用いた食品管理システムの構築とそのインターフェースの考察と評価」，明治大学理工学部情報科学科 2010 年度卒業論文，2011.

[2] 堀切　堤：「スマートにプログラミング Android 入門編第 2 版 SDK4/2.3 対応」，Tech Fun 株式会社，2012 年 6 月 11 日

[3] 間　顕次：「Andrid UI デザイン&データベースプログラミング」，ソシム株式会社，2011 年 6 月 12 日

[4] 掌田　津耶乃：「みるみるスマートフォンプログラミングがわかる本」，秀和システム株式会社，2011 年 10 月 1 日

[5] 使い勝手の法則，５つのユーザビリティ特性とは
http://www.marketingvoice.jp/columns/column/090202-01.php