

M-048

## 雛型を用いた携帯電話向け User Interface システムの開発

Development mobile phone user interfaces system based on templates

中西 正洋† Masahiro Nakanishi 小野 修一郎† Shuichiro Ono 尾上 孝雄‡ Takao Onoyo

## 1. まえがき

携帯電話端末はパーソナル志向が強い機器であり、個々の好みや利用用途に応じた User Interface(UI)のカスタマイズ要望が強い。そのため、UIはコンテンツとして携帯電話のシステム実装から分離され、コンテンツを入れ替えることで、ユーザは趣味嗜好や利用用途に応じて、UIをカスタマイズ可能となっている。

UIは、プログラム要素とデザイン要素で構成されているが、デザイナーのイメージを技術仕様落とし、処理能力の低い携帯電話上でストレスなく利用可能な UI を実現することは難しい。そこで、完成度の高い UI を作成するためには、デザイナーと開発者の地道な連携作業が必要となっている。

一方、ユーザの選択肢を多く用意し、カスタマイズ範囲を広げるためには、予め携帯電話内に多くの UI を用意しておく必要がある。しかし、一般的に UI コンテンツはデータ量が多く、携帯電話に一度に内蔵できる数に制限が加わる。

このため、大量の UI のバリエーションを提供するためには、UI の作成コスト削減と、UI の使用データ量節約が必要である。

そこで、UI の記述要素を開発者が得意な要素と、デザイナーが得意な要素に分離し開発を行い、組み合わせて UI を実現するアプローチによって、低コストで、データ使用量を節約し、ユーザに多彩な UI バリエーションを提供可能な UI システムを開発する。

開発した UI システムにおいて、実現可能な UI を紹介し、データ使用量削減の効果を示す。

## 2. 従来の UI システム

これまで、ソフトウェアの UI を別の UI に用意に切り替えるために、様々な UI システムが研究開発されてきた。これらのシステムは、ユーザインタフェース管理システム (UIMS) と呼ばれている。

古くは、アプリケーション本体から UI 描画を行うライブラリを利用する手法で、UI とシステム実装の分離が行われてきた。1990 年代に入ると、各 UI 部品をオブジェクトと見たてるオブジェクト指向のアイデアを用いて、UI とシステム実装の分離が行われており、UI はコンテンツとして分離/流通させることも可能となった。2000 年に入ると、携帯電話端末でも処理能力の向上により、ユーザカスタマイズできる UIMS として、Adobe 社[1]の Flash Lite が使われるようになった。オーサリングツールとして、市販されている Adobe Flash CS3 Professional を利用する。

† シャープ株式会社, SHARP Corporation

‡ 大阪大学, Osaka University

図1では、デザイナーがオーサリングツールを用いて UI コンテンツを作成し、携帯電話の実機上で動作確認を行い、発見した問題の修正作業を繰り返す流れを示している。

ここで、UI コンテンツは、以下の二つの要素が合わさって実現されている。

## (1) プログラム要素

イベントによる選択対象変更処理

イベントによるアニメーションの動作開始/終了処理

## (2) デザイン要素

アイコンや背景などのリソース画像

アニメーションなどの動き

UI コンテンツの作成が一通り完了すると、携帯電話で動作確認が行われるが、主に次の2点の問題が見つかる。

## ・操作レスポンスに関して

デザイナーは高速な PC を用いて UI コンテンツを作成し、動作検証を行う。デザイナーは、見た目を重視するため、描画負荷が高い視覚効果を駆使し、多数の描画要素を用いる傾向にある。しかし、PC 上で問題なく動作していても、携帯電話の処理能力は低いため、操作レスポンスが悪い UI となってしまう、コンテンツの修正が必要となる。しかしながら、操作レスポンスの良い/悪いは主観に大きく左右される。また、操作レスポンスを改善するためには、単純に描画要素を削減するだけではなくて、UI のデザインコンセプトの変更/修正も必要となる場合がある。デザインの新規性や斬新さをベースとするデザイナーと、定量的な応答速度値をベースに UI の簡略化を迫る開発者との間で、議論が続く。

## ・動作異常に関して

デザイナーはプログラムの専門家でないため、プログラム部分のコーディング誤りによる動作不具合が多発する。プログラムも得意なデザイナーが修正したり、開発者が支援したりして、不具合を修正していくが、一般にデザインも開発も得意な作業者は数が少ない。

このように、実際の UI の開発では、UI を作るために、多くの時間/費用のコストがかかっている。

また、このような手法で開発される UI は、UI として必要な記述やリソースを全て含んでおり、単体で独立した UI コンテンツとして存在している。携帯電話で UI コンテンツを入手するためには、ネットワークからダウンロードしたり、外部記憶デバイスから取得する。そのため、リソースの厳しい携帯電話下では、多数の UI バリエーションを保持することができない。

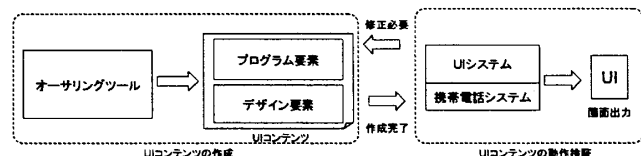


図1: 従来のUIシステムでの開発の流れ

つまり、携帯電話に UI のバリエーションを数多く提供するためには、既存の UI システムには、次の問題が存在する。

- (1) UI の開発コストが高い
- (2) UI コンテンツのデータ量が大きい

### 3. 雛型を用いた携帯電話向け UI システム

近年、Web 技術との親和性や、コンテンツとして再利用性を満たすために、UI を XML 言語で記述するアプローチがとられている[5]。しかし、XML 言語を処理する負荷は高いため、一部で利用されることもあるが、Flash メニューや、従来のシステム実装と一体化した UI を置き換えるには至っていない。そこで我々は、2D ベクターグラフィックを記述するための XML 言語である SVG 言語を利用し、その実用化を目指した。その結果、携帯電話の UI のカスタマイズというユースケースに合致し、処理能力の低い携帯電話でもストレスなく動作する UIMS を世界で初めて実用化した。

実用化の課題は、大きく分けて UI の配信システム時の課題と再生時の課題の2種類がある。今回は、UI 再生時の課題のひとつである使用メモリ量削減に関して、動作の仕組みを説明しつつ、その削減効果を示す。

#### 3. 1. 開発コストの削減

開発コストを削減するために、デザイナーと開発者が連携作業する必要性を減らし、お互いが得意な部分に専念できることを目指し、UI を利用時に UI システム上で合成するというアプローチをとる。

前述のように、UI コンテンツには、プログラムの要素と、デザインの要素に分かれる。明確に分かれているわけではなく、例えば、矩形が移動する場合、その座標値の軌跡の

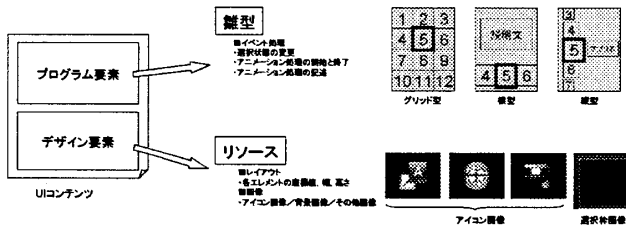


図2: UI記述での「雛型」と「リソース」

```
<?xml version="1.0" encoding="UTF-8"?>
<svg id="mySvg" width="480" height="754" xmlns="http://www.w3.org/2000/svg"
xmlns:ev="http://www.w3.org/2001/xml-events" xmlns:xlink="http://www.w3.org/1999/xlink">
  <image id="icon0101" height="H01" width="W01" xlink:href="icon0101.png" y="Y01" x="X01"/>
  ...
  <image id="icon1201" height="H12" width="W12" xlink:href="icon1201.png" y="Y12" x="X12"/>
  <text id="text0101" height="TH01" width="TW01" xlink:href="icon0101.png" y="TY01" x="TX01"/>
  ...
  <text id="text1201" height="TH12" width="TW12" xlink:href="icon1201.png" y="TY12" x="TX12"/>
  <ev:listener ev:event="load" ev:observer="#mySvg" ev:handler="#myHandler" />
  <handler id="myHandler" type="application/ecmascript">
    var state;
    function handleEvent()
    {
      evtData = evt.userData;
      prevState = state;
      if (evtData == 0) {
        /* initialize */
      }
      if (evtData == KEY_UP) {
        /* up key */
      }
      if (evtData == KEY_RIGHT) {
        /* right key */
      }
      ...
    }
  </handler>
</svg>
```

図3: 「雛型」詳細記述例

制御は、プログラム要素でもあり、デザイン要素でもある。本 UI システムでは、プログラム要素を「雛型」と呼び、開発者が作成し、デザイン要素を「リソース」と呼び、デザイナーが作成すると定義する(図2)。「雛型」は、リソースがはめ込まれる矩形と、イベントに応じた振る舞いの処理が記述されている。

図2では、「雛型」の概念図として、グリッド/縦型/横型の雛型を図示している。ここで、レイアウトも雛型で決定されるように思えるが、リソースがはめ込まれる矩形の座標値はリソースで定義される。

「雛型」の記述言語には、W3C で標準化されている SVGT1.1[2]と ECMA Script を利用する。図3に、詳細記述例を示す。スクリプト部でイベント処理を行い、μDOM 仕様を用いて、SVGT で記載されたアニメーションの起動/終了を行う。

#### 3. 1. データ量の削減

データ量を削減するために、デザイナーと開発者が別々に作成した「雛型」と「リソース」を、組み合わせて UI を生成する。生成のタイミングは UI の表示するタイミングである。図4では、3種類の雛型(A,B,C)と3種類のリソース(a, b, c)を組み合わせて、9種類の UI のバリエーションを生成していることを図示している。

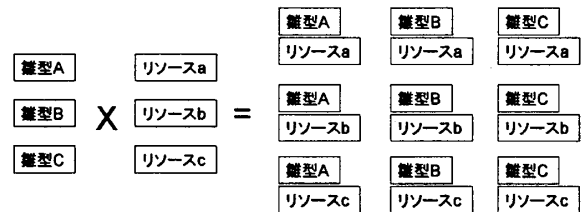


図4: 「雛型」と「リソース」を組み合わせたUIの自動生成

### 4. 計測

データ量削減の効果を計測するために実験を行った。実験環境を図5で示す。本実験のために携帯電話の内部状態を出力するツールを開発し、PC と携帯電話を USB ケーブルで接続し、携帯電話の内部状態をモニタリングした。実験用コンテンツは、実際に搭載されている図6に示す3種類(グリッド型、縦型、横型)を用意した。

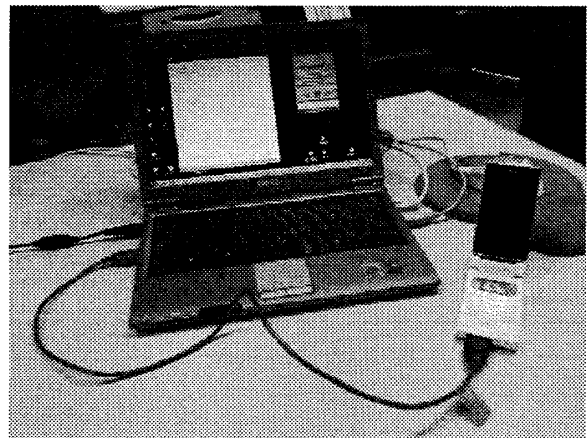


図5: 実験風景

測定結果を表1で示す。9種類のコンテンツを全て独立して保持するためには、1398294(byte)のデータが必要であったが、466098(byte)に改善された。データ量は元と比較して、約33.3%になり、リソース制限の厳しい携帯電話では大きな効果があった。

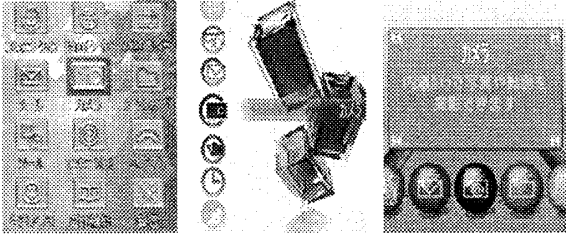


図6: 実験UIコンテンツ

表1: 測定結果

	byte		byte
雛型A(グリッド型)	18214	雛型A(グリッド型)+リソースA	124800
雛型B(横型)	40405	雛型B(横型)+リソースA	146991
雛型C(横型)	31277	雛型C(横型)+リソースA	137853
リソースA	106586	雛型A(グリッド型)+リソースB	152074
リソースB	133860	雛型B(横型)+リソースB	174265
リソースC	135756	雛型C(横型)+リソースB	164137
合計	466098	雛型A(グリッド型)+リソースC	153970
		雛型B(横型)+リソースC	176151
		雛型C(横型)+リソースC	167033
		合計	1398294

雛型とリソースのデータサイズ

雛型とリソースを組み合わせた場合のデータサイズ

### 5. 応用例

図7は、待ち受け画面での応用例である。携帯電話における端末情報とWeb情報の混在表示に関しては、[4]を参照いただきたい。

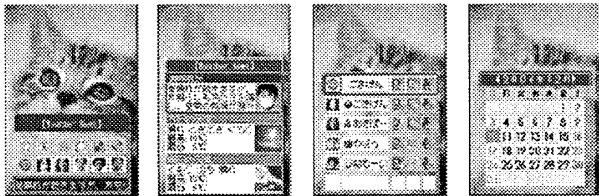


図7: 応用例



図8: 提供画面

### 6. おわりに

本論文では、新しい携帯電話のUIシステムを紹介した。本UIシステムは、UI開発において、開発者とデザイナーの役割を明確化することで、開発コストをさげつつ、厳しいリソース条件下でも、UIの利用データ量を削減可能なUIシステムである。

本UIシステムは、国内携帯電話1000万台以上に搭載されている。低コストでUIバリエーションを数多く提供できるため、1000種類を超えるUIバリエーションがネットワーク上で提供されている[3]。

また、UIコンテンツの提供Webサイトの画面を図8で示す。

さらに、本UIシステムは、携帯電話以外の商品にも数多く搭載されており、ユーザにUIをカスタマイズするという楽しみを与えている。

今後は、本UIシステムを2DUIと比較して、作成コストが非常に大きい3DUIに応用し、3DUIでも簡単に低コストでUIを作成できるシステムを研究開発していく。

### 参考文献

- [1] Adobe社ホームページ: <http://www.adobe.com/>  
1) European Computer Manufacturers Association (ECMA): ECMAScript Language Specification, <http://www.ecma-international.org/publications/standards/Ecma-262.htm> (Accessed 15 March 2007).
- [2] The World Wide Web Consortium (W3C): Scalable Vector Graphics (SVG), <http://www.w3.org/Graphics/SVG/> (Accessed 15 March 2007).
- [3] カスタモサイト: <http://www.custamo.com/pc/910/index.html>
- [4] M. Nakanishi, K. Sakakura, H. Zaima, N. Hatayama, and S. Ono, "Improvement of User Experiences for Embedded devices," Sharp Technical Journal, Sharp Co., vol. 95, no. 27, pp.58-62, Feb. 2007.
- [5] 竹居, "組込みソフトをXMLが変える 開発率向上の切り札に", 日系エレクトロニクス, 日経BP社, pp.51-58 (2005)