4L-6

携帯電話向け Java プロファイル変換

安藤　成良　　安井　浩之　　松山　実
武蔵工業大学

はじめに

多くの携帯電話で Java アプリケーションが動作するが，キャリアにより採用プロファイルが異なるため，移植作業が必要な状況である．そこで，あるプロファイル向けに作られた Java プログラムのソースを他のプロファイル向けのソースへ変換するための変換スクリプトを提案し，変換システムを開発した．本報告では，変換システムの概要および，既存ソースの変換結果に対する評価について述べる．

１．概要

プロファイルとはクラスライブラリであり，携帯電話向けには大きく分けて NTT DoCoMo が採用する DoJa，それ以外のキャリアの MIDP が存在する．アプリケーションを両プロファイルに対応させるには移植作業が必要となる．

既存の移植法には，1)プロファイル別に開発する方法，2)プロファイルに依存する部分をファイルに分け，入れ替える方法，3)プリプロセッサの利用やラッパークラスやメソッドなどの作成する方法があるが，これらは開発単位が増えたり，ソースの管理が面倒であったりする．また 3 の方法では，変換記述とソースコードが同時に書かれているため，可読性が下がる上，コードの追加による共通化をすることでプログラムサイズを増やしてしまうため，プログラムサイズの制限が厳しい携帯電話のプラットフォーム向きとは言えない．そこで，ソースコード変換をスクリプト化することによって変換記述をソースプログラムと分離する手法を提案する．これにより，散在するプリプロセッサの変換記述よりも可読性が高まり，スクリプトの再利用も可能となる．

２．プロファイルの違い

プロファイルの違いとして 1) 起動クラスの親クラス，2) 低レベル UI，3) グラフィックコンテキスト，4) 高レベル UI，5) キーイベントの処理，6) マルチメディア処理があるが，各プロファイルの持つ機能のほとんどは同じである．しかし，単純に名前や引数の置き換えで変換できるものだけでなく，オブジェクトの利用方法や，処理の違いは単なる文字列の置き換えによる変換では難しいため，ソースプログラムの解析情報を利用する必要がある．

Conversion of Java profiles for cellar phone
Shigeyoshi Ando・ Hiroyuki Yasui・ Minoru Matsuyama
Musashi Institute Technology

３．変換システム

Fig. 1 に変換システムの流れを示す．

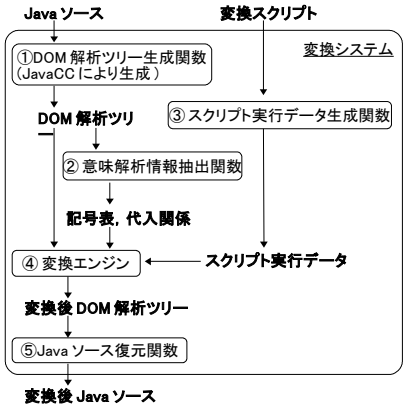

Fig. 1　変換システムの流れ

① 変換前 Java ソースを JavaCC(Java Compiler Compiler)で生成された DOM 解析ツリー生成関数により字句解析，構文解析し，解析情報を持つ DOM 解析ツリーを生成する．

変換に用いる解析情報は次のものである．

字句解析：ソースコードを意味のあるトークンに分割する．

構文解析：プログラムの位置によってトークンの文法上の意味を解析する．

DOM 解析ツリーとは XML ドキュメントで表現され，DOM(Document Object Model)を用いているため，このように呼んでいる．

DOM 解析ツリーは構文解析木を抽象化しており，テキスト要素(タグに囲まれている単なるテキスト)は持たず，要素と属性のみで構成されている．要素は Class，Method，Field などの定義要素や If や While などの制御文要素，代入文やメソッド起動式を表す StatementExpression 要素などである．例えば，Class 要素は名前や修飾子や親

クラスなどの属性を持ち，メンバ要素を入れ子にして持つ．Method 要素は Param 要素や中に記述される制御文要素や StatementExpression 要素を持ち，制御文は StatementExpression 要素と Condition 要素を入れ子にして持つ．Fig. 2 に DOM 解析ツリーの例を示す．

```
Class(name・access・superClass・ …)
 ├Field(type・access)－Initializer－…
 └Method(returnType ・access)
  ├Param(type ・name)
  ├While
  │ ├Condition－Expression(value)－…
  │ └StatementExpression－…
  └StatementExpression－…
```

Fig. 2　一般的なクラスの DOM 解析ツリー

② DOM 解析ツリーから意味解析情報抽出関数を用いて記号表，代入関係情報の抽出を行い意味解析を行う．

記号表は一般的なコンパイラで作成されるもの以外に式などの情報を含み，スクリプトによる変換対象の選択で利用される．また，変換時の名前の解決，名前衝突の回避，スコープの観測可能性とアクセス可能性の調査に利用する．記号表のデータ形式は次のようになる．

名前空間，変数名，ID，型，スコープ，修飾子

代入関係情報は代入される変数と代入するステートメントの関係をまとめたものであり，スクリプトによる変換対象の選択に利用する．

③スクリプト実行データ生成関数により変換スクリプトからスクリプト実行データを生成する．

変換スクリプトは大きく分けて絞り込み選択，操作，定義の３つの機能に分けられる．

- 絞り込み選択：定義の選択，ステートメントの選択，ステートメント位置の絞り込み
- 操作：ソースへの追加，削除，ステートメントテンプレートの抜き出し，追加，変更など
- 定義：絞込条件定義，ステートメント格納用変数定義，ステートメントテンプレート定義 (柔軟なステートメントをスクリプト上で作成するもの)

④変換エンジンは意味解析情報を参照しながら，スクリプトデータを実行し，DOM 解析ツリーを変換する．

⑤変換後の DOM 解析ツリーを復元関数により Java ソースへ復元する．

復元関数では，各要素ごとにソースを出力し，全ての要素を出力する．

４．　評価

変換スクリプトを作成し，MIDP-DoJa 間での変換テストを行った．また，プログラムサイズ，CASE ツールについても評価を行った．

４．１　MIDP-DoJa 変換

ウェブ上[1,2]で公開されている 10 個のプログラムを変換した結果，全てのソースファイルの変換に成功し，エミュレータ上での実行も確認できた．

４．２　プログラムサイズ

自作の DoJa 向けアプリケーションのソース(700 行)を 2 つの方法で MIDP 向けに変換し，プログラムサイズを比較した．1)グラフィックコンテキストをプリプロセッサによるラッパークラスを作成し，変換．2)本研究の変換システムによるソース記述の変換．結果を Table1 に示す．クラスを減らし，プログラムサイズを約 1KB を節約することができた．これにより，画像や音のメディアリソースに削減できたサイズをまわすことができる．

Table 1　変換後プログラムサイズ

ラッパー利用	変換システム
11550Byte	10510Byte

まとめ

提案手法を用いて開発した変換システムにより，DoJa-MIDP 間での変換が可能であり，ソースファイルと変換記述を分離した結果，既存の移植法よりもプログラムサイズを小さくすることができた．この変換システムは携帯電話向け Java アプリケーションに限らず変換が可能であり，ファクタリングツール・Java ソース解析ツールとしての使い方も考えられる．

また，変換スクリプトの作成や変換の実行を，より容易に行えるような CASE ツールを開発しており，このツールによりスクリプト処理系の利便性を高められると考えている．

参考文献

[1]Appli-Style.com：
http://appli-style.com/doja_sample/index.html
[2]Java・i アプリ入門：
http://www.geocities.jp/java_iappli/java9.htm