J2MEによる制御システムの開発事例

デモ－7

西山 裕二　佐藤 洋介　松永 武　安井 浩之　松山 実
武蔵工業大学

1. はじめに

現在，携帯電話やPDA，その他いろいろな組み込み機器において，組み込み機器向けのjavaであるJ2ME(Java 2 Micro Edition)[1]が使用されるようになってきた．しかし，実際の開発現場では新技術に対応しない従来の開発プロセスである「ハードウェア中心，ソフトウェア後発」の体制を採用しているところが多い．このような開発プロセスを用いてのシステム開発では，製品の生産性を向上させることは難しい．また，システム構成部品の多機能化により，プロジェクトチーム内での開発システムに対する相互理解がより重要視されるようになっている．

このような背景により，組み込みソフトウェア開発では，新技術の投入と同じように，開発プロセスの改善も必要である．

本開発事例では，オブジェクト指向プロセスとUML(Unified Modeling Language)を使用して分析設計を行い，J2ME_CLDC(Connected，Limited Device Configuration)[2]で実装，動作するロボットを開発した．この開発事例を通して，J2MEによる組み込みソフト開発に，オブジェクト指向開発プロセスを適用したときのノウハウ及び有用性について報告する．

2. 開発事例

2.1 システム概要

本制御システムは，PalmOSを搭載したPDA(WorkPad50J)を用いて模型戦車のモータ制御を行うシステムである．この戦車及びそのシステムは図1に示される階層モデルで構築されている．それぞれの階層の役割を以下に示す．

＜＜GUI 行動命令パターン＞＞

本システムはあらかじめ与えられた複数の行動プログラム(行動命令)を順次実行する．この行動命令には動作名，動作時間，方向の三つのデータが入っている．戦車はこの命令に従って，動作を実行する．行動命令の集合を行動命令パターンと呼ぶ．

＜＜戦車フレームワーク＞＞

分析で抽出されたオブジェクトの実装である．一つ上の階層の行動命令，行動命令パターンはインスタンスであり，そのクラスが実際に定義されているのがこの階層である．

＜＜CLDC API＞＞

使用した開発環境及び，実行環境はJ2ME_CLDC．

＜＜PIC ドライバ＞＞

行動命令は，PDAとPIC[3]をシリアル接続して，順次PDAから命令を送ることによって，戦車を動作させる．モータはPICにより制御される．

＜＜ハードウェア＞＞

実際の回路，モータ，キャタピラなど

GUI 行動命令パターン
戦車フレームワーク
CLDC API
PIC ドライバ
ハードウェア

図1 階層モデル

2.2 開発プロセス

本事例では“OCTOPUS”[4]をリファレンスモデルとする開発プロセスを採用し，組み込みシステム特有の問題点であるハードとソフトの開発サイクルの違いについてはサブシステム分割，インクリメンタルアプローチによって対処した．サブシステムは「Coadのコンポーネント」[5]を基に分割を行いハードウェアとソフトウェアの開発を並行して行うことを可能にした[6]．

2.3 サブシステム分割

Coadのコンポーネントに基づき，アーキテク

Development of control system with J2ME.
Yuji NISHIYAMA,Yosuke SATO ,Takeshi Matsunaga,
Hiroyuki YASUI, Minoru MATSUYAMA
Musashi Institute of Technology

チャ設計において階層モデルを図２のようにサブシステムに分割した.

図２　システムアーキテクチャ

HI(Human Interface)は GUI のオブジェクトで構成されていて，システム利用者と PD を仲介する役割を持っている．PD(Problem Domain)は分析段階で抽出されたオブジェクトで構成される．本システムでは階層モデルの戦車フレームワークが PD のオブジェクトとなる．SI(System Interaction)は外部システム，すなわちハードウェアとの仲介を行うオブジェクトである.

HI/PD/SI と分割することにより，各サブシステムのインタフェースを事前に協議しておけば，次の詳細設計ではそれぞれ並行して開発を進めることができる.

2.4　J2ME 実装を考慮した詳細設計

今回，J2ME_CLDC を用いる際に，システム仕様として求められた技術として，シリアル制御がある．また本事例で使用した J2ME は，主に PalmOS 向けの実行環境であり，CLDC API とプロファイルとして KJava API を含んでいる.

シリアル制御については，CLDC API の GenericConnection という入出力フレームワークを用いることにより実現した．このフレームワークはメモリ領域の小さい J2ME の入出力のクラスを一般化したものである．これにより，J2SE(Java 2 Standard Edition)実装に対して簡単にマッピングが可能で，かつ新しいデバイス及びプロトコルを追加する際の拡張性，柔軟性，及び整合性を保持できる．本システムのシリアル制御処理は，図２のサブシステム SI が他のサブシステムに対して処理を隠蔽している.

GUI については，主に PalmOS で用いる部品を備えた KJava API を用いて実装した．KJava API は MVC アーキテクチャを採用していないので，GUI 設計において標準では View と Controller の責務を分けることができない．本事例では，一つの方法例を紹介するため，それぞれをクラスとして分割した．クラスの構成は STATE パターンを適用させて実現した．これらの GUI に関するクラス群はサブシステム HI に分類されている.

分析工程で抽出された問題領域のクラス群はサブシステム PD に分類されている.

3. まとめ

J2ME を用いる際にサブシステムに分割するメリットとして，基本ライブラリを用いた実装と，プロファイルを用いた実装を分割できることが挙げられる．本事例では PD と SI は CLDC API のみで実装されていて，KJava API を用いているのは HI だけである．例えばプロファイル MIDP を用いている携帯電話などでこのシステムを使用したい場合，HI を入れ替えるだけでよいことになる.

オブジェクト指向を用いたメリットとしては，ドメイン分析を十分に行うことできたため，次工程において致命的な開発ミスが起こらなかったことである．また統合開発プロセスを用いたことにより，サブシステムを並行して開発できたので全体的な計画の遅れが生じなかった.

以上より，オブジェクト指向開発プロセスを，J2ME によるソフト開発に使用することは，有用であると確認できた.

参考資料

[1]Java™ 2 Platform, Micro Edition
http://java.sun.com/j2me/
[2]CLDC and the K Virtual Machine (KVM)
http://java.sun.com/products/cldc/
[3]後閑　哲也：電子工作のための PIC 活用ガイドブック，株式会社技術評論社(2000)
[4]OCTOPUS
http://www-nrc.nokia.com/octopus/
[5] 中谷多哉子，青山幹雄，佐藤啓太　他　：bit 別冊　ソフトウェアパターン，11 月号，共立出版株式会社(1999)
[6]佐藤　洋介，他　：UML を用いた制御システムの開発事例，第 63 回情報処理学会全国大会 3Q-04(2001)