5U−4

金融機器組み込みソフト向けPOLの開発*

藤巻 昇　清水 洋子　岡安 二郎　田中 利幸　小尾 俊之†

株式会社 東芝 システム・ソフトウェア技術研究所‡

1 はじめに

現在、ソフトウェアの高生産性、高品質の確保が求められる中、対象分野に特化した開発手法やツールにより支援していくドメインCASEの効果が期待されている。

我々は、ATM(Automatic Teller Machine)制御用の組み込みソフトに対する、POL(Problem Oriented Language)の開発を行っている。ATM制御ソフトの開発は、基本的なソフトが存在し、これを顧客毎の仕様対応に変更することで行われ、新規に作成されることは少ない。そこで、顧客毎の仕様の違いをソフトウェアに容易に反映し、ソフトの一部を自動生成することにより、開発期間の短期化、生産性の向上を実現することを目的としている。

以下、仕様記述モデルに焦点を当て、詳細に報告する。

2 仕様記述モデル

2.1 ATM制御ソフト構成の概要と対象範囲

既存のソフト構造は図1に概観できる。ATM制御ソフトはマルチタスクで処理を行っている。メイン制御タスク群の下で複数の処理用タスク(**取引処理タスク**)が取引の種類に応じて動作する。また、デバイス単位にある程度自律的に動作するタスク(**デバイスタスク**)が用意されている。なお、タスク間の情報交換にはメールが使用されている。

この中でも、ユーザ（銀行）毎にインデント開発が必要な取引処理部分(図1斜線部分)をPOL化の対象として取り上げることにした。これは自動生成による効果がもっとも期待できるからである。

2.2 ATMソフトの特徴と仕様記述の目標

仕様記述モデルを決定するにあたり、既存のソフトウェアを以下の2つの観点から分析することにした。

1. 自動生成されたソフトは、対象外の既存のソフトと共存する必要がある。特にデバイスタスクはそのまま用いたい。そこで、仕様記述モデルを決めるにあたって、対象部分のATMソフトの特徴を、対象外との関係に着目して分析した。
2. 現状の開発方法では、顧客毎の仕様対応に変更するのは容易でなかった。その背景には、ソフトウェアの仕様が複雑で変更範囲の特定がしづらい、という理由があった。変更範囲の特定が容易でないのは対象部分のどういった特性によるものなのか、という点に着目し分析した。

上述した観点でのATMソフトの特徴と、対象範囲における仕様記述能力の主な目標は以下の2点にまとめられる。

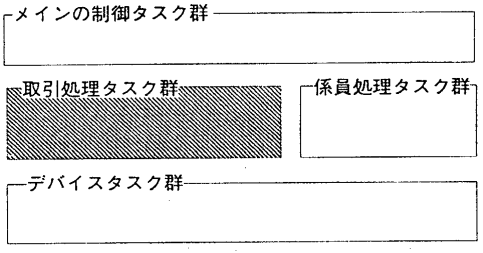

図1: ATMソフト構造

1. メールドリブンな構造

   デバイスタスクは、取引処理タスクからの1つの命令によりある程度独自に一連の処理を進め、その途中経過を逐次メールで知らせてくる。従って取引処理では、伝送されてくるメールを調べ処理を実行していく。つまり、処理進行のきっかけはメールの送受信である。

   この特徴を効果的に記述するには、

   1)「ある状態で受信可能なメールを明確に記述可能な仕様記述モデル」

   が必要とされる。

2. 複数のデバイスの並列性

   ATMの処理の中には取引の過程でスピード向上のために複数のデバイスを同時に動かす処理が存在する。このようにデバイスを並列に動かすときには、デバイス間の処理の同期を取るため両者の状況を調べながら処理を実行する必要がある。

   現状の仕様は処理を切口として作成されているので、並行動作させるデバイスの処理の組み合わせ毎に異なる仕様を作成し、組み合わせの少しの違いに応じて類似した仕様が複数できるという冗長さを含んでいた。この方法では、デバイスの処理に仕様の変更が生じた場合、変更箇所が複数の仕様上に散らばり、変更範囲が特定しづらいものとなる。

   このことから、デバイスの切口からの仕様が必要となる。そこで、

   2)「処理の切口とデバイスの切口を融合し変更範囲を集中させ、冗長さなく記述可能な仕様記述モデル」

   が必要である。

---

*Development of a POL for Software embedded Automatic Teller Machine

†Noboru Fujimaki, Yoko Shimizu, Jiro Okayasu, Toshiyuki Tanaka, Toshiyuki Obi

‡Systems & Software Engineering Lab. TOSHIBA Corp.

2.3 仕様記述モデルの決定

前述の特徴を踏まえて以下のように仕様記述モデルの決定を行った。

1. 状態遷移モデルをベース
   ある状況での受信可能なメールを表現した仕様書は存在したが、処理の流れが追いづらかった。そこで上記1)の要求を満たし、かつ処理の流れのtraceabilityを上げるため、仕様記述方式のベースとして状態遷移モデルを採用した。これは、メールドリブンな構造という特徴と状態遷移モデルの親和性が良いからである。メール待ちを状態、待っているメールをイベント、実際の処理をアクションに対応づけることで状態遷移モデルにより記述可能である。

2. 並行動作からインタラクション部を切り分ける
   並列処理の記述による冗長さは、デバイス毎の処理とデバイス間の関係の処理を仕様中に混在させたことに起因する。この結果デバイスの組み合わせ毎に違った仕様が必要になった。そこでこの問題を解決するために、デバイス間のインタラクションに関する処理を切り分けることにした。

   仕様記述は図2の様に、各デバイスタスクの動作を監視しながら、独自にできる処理を実行する「オブジェクトモニタ」と、それらオブジェクトモニタ間のインタラクションに関する処理を行う「メインモニタ」に分割して記述する。オブジェクトモニタはデバイスタスクに対応して1つ、メインモニタは取引ごとに1つづつ用意する。これにより処理の切口とデバイスの切口を融合する。

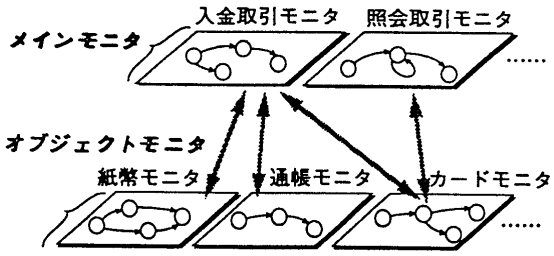

図 2: 仕様記述モデル

以下に簡単な例を用いて(図3)上記の仕様記述を説明する。

この仕様はデバイスタスクAとBを並列に処理する取引の制御を記述した例である。AとBは途中で同期を取る必要がある仕様だと仮定する。このときAモニタはデバイスタスクAを監視し、Bモニタはデバイスタスクを監視する。メインモニタには、デバイスタスクA及びBのインタラクションに関する処理を記述している。

モニタの動きは以下のようになる。それぞれのオブジェクトモニタ(A及びBモニタ)はデバイスタスクからの送信メールにより状態を遷移させる。デバイスAとデバイスBの同期が必要な状態(図中斜線部分)でそれぞれメインモニタに同期が必要なことをメールで知らせる。メインモニタは、両モニタからの同期の要求メールを受信完了した時点でA及びBモニタに処理再開のメールを送信する。これを受けたそれぞれのモニタは、デバイスタスクA及びBに対してそれぞれ処理の再開を指示する。このようにデバイスタスクA及びBの並列動作を制御する。

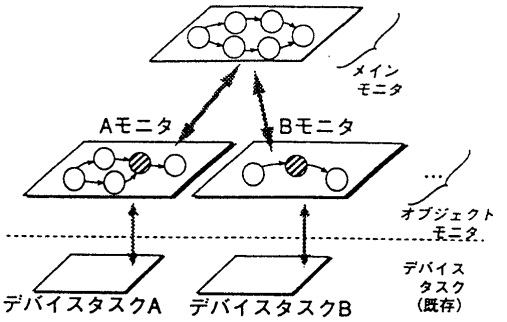

図 3: 仕様記述例

3 評価

現在、このモデルに則った仕様記述言語を用いて実際の仕様を記述することにより、記述モデルの評価を行っている。以下に2つの観点から評価を述べる。

1. 状態遷移ベースの仕様記述
   作業者が与えられるSEからの要求仕様書には、状態遷移指向の自然言語記述があり、このモデルは作業者にとっても受け入れ易い。また、状態遷移図を使用することで、処理の流れを視覚的に捉えることができ、作画によって容易に仕様の変更が行えるなどの利点がある。

2. デバイス間のインタラクション部の切り分け
   インタラクション部を切り離したことにより、各デバイスに依存した記述部分は各取引で共通化することが可能となる。また顧客毎の仕様の違いはインタラクション部分に集中する傾向があるので、仕様の再利用率が向上することにもなる。

4 おわりに

現在、POL化の対象取引処理の仕様を状態遷移モデルで記述しながら評価を継続中である。開発に際しては、厳密な記述、理解容易性、変更容易性、既存ソフトウェアとの共存を目標として進めてきたが、本稿で提案した「拡張状態遷移モデル」と「インタラクション部を切り分けたソフト構造」の採用で、これらの目標は達成できた。また、ソースコードの生成方式、仕様検証方式を含めたCASE環境の検討も行っている。今後は現場での適用を行いこのモデルの有効性を評価する。

参考文献

[1] 加地 他, 状態遷移をベースとした部品合成システムの試作, 情報処理学会第38回全国大会予稿集, pp1199-1200(1989)

[2] 岸 他, 状態遷移モデルに基づくプログラム部品合成システムの開発, 情報処理学会 ソフトウェア工学研究会, Vol.91, No.66, 80-22(1991)