CafeOBJ を用いた在庫管理問題の記述と評価

坂本 淳誌 (s0810027)

北陸先端科学技術大学院大学 情報科学研究科

2011 年 2 月 8 日

キーワード: フォーマルメソッド, CafeOBJ, ラピッドプロトタイピング, C 言語, 在庫管理問題.

ソフトウェアの開発現場において、仕様に関するトラブルが多く発生している。これらの対応として、仕様を数学的に記述・分析することで品質の高いシステムを効率よく開発する手法であるフォーマルメソッドに関心が向けられている。このフォーマルメソッドの導入事例として、FeliCa の IC チップのファームウェアが挙げられる。FeliCa は使用されている分野が非常に多岐にわたっており、不具合が存在することはあってはならない。この FeliCa を携帯電話に不具合なく組み込むため、仕様記述言語である VDM++での仕様の記述及びテストを行い、FeliCa IC チップのファームウェアの作成を行った。ファームウェア作成後の調査において、仕様の理解不足と見落としを除き、仕様に関しての不具合原因が非常に小さいことが報告されている。CafeOBJ は VDM++同様に形式仕様言語であり、代数に基づく実行可能な形式手法言語である。しかし、現在までに CafeOBJ で記述された仕様からプログラムが実装された例はない。そこで本論文では、形式手法言語 CafeOBJ でラピッドプロトタイピングを行い仕様を記述する。この仕様を仕様書としてプログラムを作成し、そのときに得られた効果や影響を評価・考察する。ラピッドプロトタイピングは問題をプログラムで試作する手法であり、仕様について早期に議論し、完成に近い仕様を作成することができる。

本論文では、対象とする問題を在庫管理問題とした。在庫管理問題は情報処理学会で出題された、プログラムの設計技法に関する問題である。この問題に関して、数多くの設計技法の研究がされているが、これらの研究では仕様を形式的に記述するか、ラピッドプロトタイピングを行っているのみである。要求に従って仕様書を作成する段階からプログラムを実際に作成することで、CafeOBJ が開発において与える効果などをより実際の開発に近い形で評価・考察することができる。また、実際の開発に近づけるため、実装を行うプログラム言語は採用している分野が幅広い C 言語とした。

CafeOBJ で仕様を記述する前に、対象とする在庫管理問題について問題の要求を深く理解をする必要がある。本論文では問題理解のために在庫管理問題のデータ構造とその手

順について、UML 図からクラス図とアクティビティ図を採用した。この 2 つの UML 図を作成することで、手順と求める要求について理解に努めた。

この UML 図と問題の仕様を基に CafeOBJ で仕様記述を行った。その結果、CafeOBJ で仕様の記述を行った際に、アクティビティ図に変更があった。この変更点はプログラムの手順を合理的にするものであり、UML 図で記述を行っている際には気付けなかったことである。これは対象とする問題の理解をより深める結果となったと言える。

次にこの CafeOBJ の仕様を仕様書とし、C 言語で実装を行う。この実装において、可能な限り仕様書に近い形の実装を行いたいと考え、実装の前に変換規則を提案した。提案した変換規則は、線形リストの使用、モジュールと関数の取り扱い、データ構文の 3 つである。この変換規則を用いて C 言語でプログラムの実装を行った。

ここまでの開発において、CafeOBJ で記述した仕様を仕様書としてプログラムの実装にどのような効果があったか、問題点など 6 つの項目が挙げられた。まず 1 つ目は、仕様記述を行うことで問題の理解が深まったことが確認できた。仕様記述の前に作成したアクティビティ図では出庫依頼票と積荷票の出庫指示票は別々の関数で作ることを前提としていた。しかし、仕様記述を行うことで、この 2 つの票どちらを受け取った場合でも、共通動作部が存在することがわかった。2 つ目は、汎用性の高い関数やモジュールは実装において仕様との差異を生むことである。CafeOBJ はモジュールを用いて、関数の型をある程度制限せずに使用できる。しかし、C 言語のように関数の引数を固定しなければならないプログラム言語においては引数の型をある程度自由にできるモジュールは差異を生む原因となる 3 つ目は、モジュールは細分化しないことである。C++や Java といったプログラム言語ではクラスなどにモジュールを当てはめることができる。よって、細分化させ過ぎずに機能をまとめたモジュールを記述するほうが望ましい。4 つ目は変数名には意味づけを行った名称にすることである。本論文で作成・使用した変数は一時的な、深い意味のない変数名であった。しかし、この変数名を明確にすることで関数に使用する引数や変数名を固定することができ、仕様をより厳密にすることができる。5 つ目はプログラムの記述法の規則に従うことである。開発の現場に置いて、関数名や変数名の大文字、小文字の使用法に関してルールを決めている場合が多い。大文字と小文字の違いでうまく動作しないケースも考えられるので、プログラムの命名規則に従って CafeOBJ の記述を行うことが望ましい。6 つ目は記述外の関数を作成する必要があることである。CafeOBJ 記述の仕様では記述する必要のない関数を、C 言語で実装しなければならないケースがある。これに関しては、CafeOBJ の仕様書より前の問題定義において考える必要がある。

以上のことから、CafeOBJ 記述の仕様書の作成は問題の理解に有効ではあるが、記述方法を検討する必要があると考えた。