差分概念辞書を用いた過度の仕様抽象度の修正

4M－4

落合 勝博† 松澤 裕史‡ 深澤 良彰†

†早稲田大学理工学部 ‡日本IBM(株) 東京基礎研究所

1 はじめに

ソフトウェアの開発方法の一つであるトップダウンアプローチによる仕様記述は、抽象的な仕様記述を段階的に詳細化する作業としてとらえることができる。この方法に従って記述を行なうと、それぞれ別の抽象度で記述された複数の階層的な仕様が得られる。各階層においては、その中の記述が同一の抽象度で記述されていることが望ましいが、それは必ずしも保証されていない。

同一の階層中に、それぞれ抽象度の異なる記述がある場合、その仕様を理解しようとする人は、一方では抽象的な記述を、また一方では具体的な記述を一度に理解しなければならない。この結果、全体的に見ればそれほど重要でない一部の具体的な記述を、抽象的に記述された部分と同様に重要なことだと勘違いをする恐れがある。これは、仕様を理解しようとする上で誤解を招く原因になる。

仕様の抽象度を均等にする手法としては、データフロー図をプロセス分解代数を用いて修正する手法[1]がある。しかし、この手法ではプロセスに入出力するデータ数によって修正を行なうため、仕様中の意味までは考慮しない。このため、修正後に、意味的な抽象度の均等性は保証されない。

本稿では、仕様の階層的な記述方法の一つであるデータフロー図を対象に、同一階層の異なる抽象度による記述を、意味を考慮して指摘・修正することで、同一階層中の抽象度を揃える手法について述べる。

2 過度の抽象度と差分概念辞書

仕様にはいくつかの概念(同義語は同じものとして考えた場合の用語)が記述されている。その仕様中で、ある概念の抽象度が他に比べて低いというのは、その概念特有の意味が仕様中の他の概念で利用されていないにもかかわらず、その概念が記述されていることであると、筆者らは考えている。これを、本研究では概念の抽象度を判断する基準にする。この判断基準に従うと、概念特有の意味が仕様中に利用されている時、その仕様は適度な抽象度で構成されていることになる。また利用されていない時、他の部分よりも具体的な抽象度(過度の抽象度)を持つということになる。

この判断基準を利用するため、概念を意味の継承関係で階層的に表した差分概念辞書を用意する。差分概念辞書に記述する項目は、概念名、意味上の上位概念、および上位概念との差分概念である。差分概念には、上位概念に対するその概念独自の意味を図1の規則で記述する。

図2は差分概念辞書の例である。この差分概念辞書は、図3中の概念「在庫不足リスト記入」に対応する。差分概念は、辞書に記述する概念名に近い順に表す。したがって、図3中の出力データフローの「不足品目」、ストアの「在庫リスト」は、差分概念辞書では、図2の順番で記述される。

差分概念辞書は次のように利用される。図2の差分概念を用いて、図3中のストア「在庫不足リスト」がなかった場合、上位の「在庫記録」の方が適切であると判断する。ここでは「在庫不足リスト」があるので、この概念は仕様中で適切な抽象度を持つと判断される。

なお、差分概念辞書は、抽象度の判断を行なう前に、あらかじめ過去の仕様を参照し作成しておく。

```
差分概念:={A[,B]./ B<,A>.}

A:=D/C
B:=P/S/T
D:=(入力/出力)データフロー::データ名
C:=(入力/出力)制御フロー::制御名
P:=プロセス::プロセス名
S:=ストア::ストア名
T:=ターミネータ::ターミネータ名
---------------
記号の意味: a/b/c/d: a～dのうち一つ
            [a]: aが0もしくは1個
            <a>: aを0個以上
            {a}: aを1個以上
```

図1: 差分概念の記述規則

```
  概念名: 在庫不足リスト記入
上位概念: 在庫記録
差分概念:
  出力データフロー::不足品目, ストア::在庫リスト.
```

図2: 「在庫不足リスト記入」の差分概念辞書構成

3 過度の抽象度の指摘手法と修正の手順

本手法では、仕様中の過度の記述部分を、差分概念辞書を利用して指摘する。ユーザはこの指摘部分を、適切と思われる上位の抽象度に書き直す。以下には、差分概

A modification method of overwritten specifications using the difference concept dictionary

Katsuhiro OCHIAI†, Hirofumi MATSUZAWA‡,
and Yoshiaki FUKAZAWA†

† School of Science & Engineering, Waseda University

‡ Tokyo Research Laboratory, IBM Japan, Ltd.

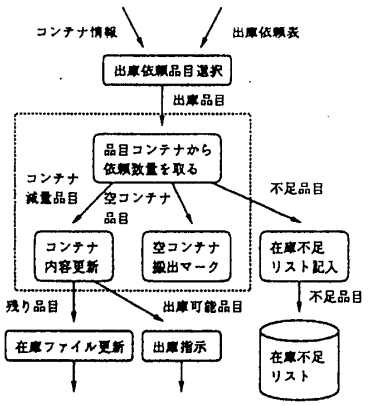

図 3: 過度の抽象度を含むデータフロー図

念辞書を利用した過度の抽象度の指摘手法と、ユーザによる修正手順を示す。

差分概念辞書を用いることで、データフロー図中の隣接する概念 (例えば、あるプロセスと、それに入出力しているデータ) 間の抽象度を調べることができる。

隣接する概念をそれぞれ A,B とした時、隣接する概念間の抽象度の高低は、差分概念辞書を用いて以下のように求められる。

- B の差分概念のみが仕様中で利用されている場合 A<B(A は B よりも抽象度が低い)
- A の差分概念のみが仕様中で利用されている場合 B<A(B は A よりも抽象度が低い)
- B も A も差分概念が仕様中に利用されている場合 A=B(A と B の抽象度は同じ)

このとき、過度の抽象度の修正を次の手順で行なう。

1. 仕様中の全概念に対して、差分概念辞書を用いて隣接する概念間の抽象度の高低を求める
2. 全概念中で、隣接するどの概念よりも抽象度の低い概念群を過度であるとしてユーザに対し指摘する
3. ユーザは、指摘された概念群に対して、上位の概念に置き換え、元の階層を書き直す
4. この指摘・修正を全概念の抽象度の高低差がなくなるまで繰り返す

4 評価

酒類販売会社の倉庫管理問題 [2](図 4) を要求文として、数人に仕様を記述してもらった。また記述の際には、特に階層的になるように留意してもらった。図 3は、その仕様の一部である。この図 3の仕様に対して本手法を適用した結果、図 3中の点線で囲まれた部分が過度の記述として指摘された。この指摘箇所をユーザが手によって修正した。その修正後の仕様を図 5に示す。

図 3では、「出庫処理」に関することを詳細に記述している。そのためデータ処理の「出力」に関する部分が分かりづらくなっている。図 5では、抽象度を揃えることによって、入力−処理−出力というデータ処理の基本構造が図 3に比べて分かりやすくなっている。

本実験では、仕様の中に記述された全ての概念に対して差分概念辞書を利用し、本手法を適用した。その結果を表 1に示す。

ある酒類販売会社に毎日数個のコンテナが搬入されてくる。受付係は毎日数 10 件の出庫依頼を受け、倉庫係へ出庫指示書を出す。在庫が無いか数量が不足の場合には、依頼者に在庫不足連絡を出し、在庫不足情報を在庫不足リストに記入する。受付係が依頼者に在庫不足連絡を出す処理をシステム化せよ。

図 4: 酒類販売会社の倉庫管理問題

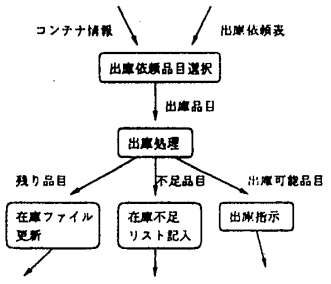

図 5: 修正後のデータフロー図

表 1: 倉庫管理問題に対する本手法の適用評価

差分概念辞書の総概念数	81
仕様中に記述された概念の総数	90
総指摘箇所数	10

試験者は階層的になるように留意して記述したにも関わらず、本手法によって、10 箇所の概念に対して過度の記述を指摘することができた。その理由として、試験者がある階層を記述する際には、記述しようとする部分の複雑さによって、記述する抽象度を変えていることがあると思われる。すなわち、記述しようとしている部分が複雑な場合には抽象的に記述し、単純な場合には具体的に記述する傾向がある。それゆえ、一つの階層中の抽象度を揃えることは難しいことがある。したがって、本手法による指摘は、従来の階層的仕様記述方法に対して、有効であると思われる。

5 まとめ

階層的な仕様記述には、異なる抽象度による記述があった場合、開発者間の誤解の要因になる場合がある。本稿では、差分概念辞書を用いることによって、必要以上に詳細な部分を指摘し、その箇所を修正することで、仕様の抽象度を揃え、この問題の解決を図る手法を提案した。

今後、より複雑で大規模な問題に対しても実験を行ない、本手法の有効性について評価を行なう予定である。

参考文献

[1] Adler,M.: An Algebra for Data Flow Diagram Process Decomposition, IEEE Trans. on SE., Vol.SE-14, No.2, pp.169-183, 1988

[2] 山崎利治: 共通問題によるプログラム設計技法解説, 情報処理, Vol.25, No.9, p.934, 1984