

形式的仕様記述におけるわかりやすいネーミング*

2D-8

若松 直樹, 清水 洋子, 玉木 裕二, 小尾 俊之†

株式会社東芝 研究開発センター システム・ソフトウェア生産技術研究所†

1.はじめに

形式的仕様記述を用いて動作仕様を定義することにより、ソースコードの自動生成や仕様検証等の支援を可能にする研究が盛んに行われている [1][2]。しかし、それら高度な支援を実現するために人間にとっての理解性を犠牲にして計算機にとって解釈が容易なものにする場合が多い。

一つのシステムを複数に分割して記述した時、各仕様からソースコードを生成する際に、仕様上で同じ名前のものをソースコード上でも同じ名前にしなければならない。そのためにはネーミングに規則を設ける必要がある。また、検証を行うために、名前の意味を解析できることは重要である。ソースコードへの変換を考慮したり、記号に意味を持たせて、仕様上の名前に付加し、それにより解析を可能にするという規則は、人間にとっての理解性を低下させる。これを改善するために、我々日本人にとって最も理解しやすい日本語の文章で仕様を記述するネーミング手法を提案する。

本報告では仕様上に記述された日本語の名前をソースコード上の名前に変換するためのネーミング規則、およびその解析アルゴリズム、また本提案による効果について、ATM（現金自動取引装置）向け CASE ツールへの適用における、特にメッセージ名を例にとって述べる。

2. ネーミング規則と解析アルゴリズム

わかりやすい日本語での記述を可能にする最も容易な方法は、日本語で記述された名前一つ一つと、それぞれの解釈を辞書に用意することである。この方法では自由に日本語を記述できるが、辞書が肥大化してしまう。そこで、メッセージに用いられる単語のみを辞書に用意することで辞書を小さくし、また、辞書に用意した情報から意味解析が可能な、人間にとっての理解性の高い日本語を記述するための規則を考える。

我々はまず、実際の ATM 向けソフトウェアの動作仕様を分析した。その結果、まずメッセージの主語は ATM としてメッセージ名には表さないことにし、また必要

*Easily Understandable Naming for Formal Specification
†Naoki WAKAMATSU, Yoko SHIMIZU, Yuji TAMAKI, Toshiyuki OBI
R & D Center, Systems & Software Engineering lab., Toshiba Corp.

となる単語を整理し、“物を表す名詞” “動作を表す名詞” “場所を表す名詞” “状態を表す名詞” “副詞” “動詞” に分類した。動詞に関しては更に、～するという指示とそれに対する～した等の返事のパターンを調べた。そしてこれらの単語を結ぶために助詞を用いることとし、単語の種類と助詞との関係を次のように定めた。

“物” = “物を表す名詞” { “と” “物を表す名詞” }
{ “の” “物を表す名詞” }

“動作” = “動作を表す名詞” { “と” “動作を表す名詞” }

“場所” = “場所を表す名詞”

“状態” = “状態を表す名詞”

“文” =

“物” (“を” | “に” | “が”) [“副詞”]
“動詞”

| “物” “を” “場所” (“へ” | “に”) [“副詞”]
“動詞”

| “場所” (“に” | “へ”) “物” “を” [“副詞”]
“動詞”

| “物” “の” “動作” (“を” | “に” | “が”)
[“副詞”] “動詞”

| [“物” “が”] “状態” (“に” | “と”)
[“副詞”] “動詞”

メッセージ = “文” { “、” “文” }

| “文” [“時に” “文”]

※ () は 1 回、[] は 0 回または 1 回、{ } は 0 回以上の繰り返しとする

以下に例を示す。

- カードの取込みを強制的に打ち切る

“物を表す名詞” の “動作を表す名詞” を “副詞” “動詞”

- 通帳を取り込む時に障害となった

“物を表す名詞” を “動詞” 時に “状態を表す名詞” と “動詞”

こうして記述したメッセージから、辞書を参照しながら単語を切り出し、その種類と並びを識別することでその内容を解析できる。これによりソースコードへの変換が行える。例えば切り出した単語を辞書を参照してソースコード上の名前に置換し、適当に言葉を補いながら変換するという方式が考えられる。その具体的なには次のような規則を定めた。

“物” = “物を表す名詞” { “AND” “物を表す名詞” }
{ “物を表す名詞” }

“動作” = “動作を表す名詞” { “AND” “動作を表す名詞” }

“場所” = “場所を表す名詞”

“状態” = “状態を表す名詞”

“文” =

“動詞” “物” [“副詞”]

| “動詞” “物” TO “場所” [“副詞”]

| “動詞” “物” TO “場所” [“副詞”]

| “動詞” TO “動作” “物” [“副詞”]

| “動詞” “状態” [“副詞”]

メッセージ = “文” { “AND” “文” }

| “文” “WHEN ATM” “文”

※単語間には “_” を挿入する

以下に例を示す。

● PREPARE_TO_RETURN_PASSBOOK

“動詞” “動作を表す名詞” “物を表す名詞”

● COMPLETED_MATCH_CARD_AND_PASSBOOK

“動詞” “物” (“物を表す名詞” と “物を表す名詞”)

ここでは辞書を小さくするために以下のような工夫をしている。まず動作を表す名詞は、不定詞を用いて表すことで、“動詞”と“動作を表す名詞”に同じ単語を登録し、また“動作を表す名詞”+“する”で動詞になるものは名詞のみを辞書に登録し、名詞から動詞を識別することで、また動詞が指示を表す場合や返事を表す場合などの活用も、指示を表す形のみを辞書に登録し、それから完了などの返事を表す形を識別することで、更に辞書の大きさを小さくしている。

3. 効果

本節では、従来と今回提案した手法を比較する。まず、今回提案した手法で記述したメッセージの実例を示す。

1. 通帳を取り込む
2. 通帳を取り込んだ
3. 通帳の取込みを開始した
4. 通帳を取り込めなかった
5. 通帳を取り込む必要がなかった
6. 硬貨口のシャッタを開ける
7. 硬貨口のシャッタを開けた
8. 硬貨口のシャッタを開け始めた
9. カードの取込みと読取りを強制的に打ち切る時に障害となった

従来は単語の羅列で表現していた。上記のメッセージを従来通り記述すると以下ようになる。

1. 通帳_取込
2. 通帳_取込 → 通帳_取込_完了
3. 通帳_取込_開始 → 通帳_取込_開始_完了
4. 通帳_取込 → 通帳_取込_不可
5. 通帳_取込_必要 → 通帳_取込_必要_無
6. 硬貨口_シャッタ_開
7. 硬貨口_シャッタ_開 → 硬貨口_シャッタ_開_完了
8. 硬貨口_シャッタ_開始 → 硬貨口_シャッタ_開始_完了

9. カード_取込_読取_強制_打切_時_障害

(矢印の右側：同名をさけ、正しい意味を持たせるために語を付加した例)

従来の手法には以下の問題点が存在していたが、今回それが改善されていることがわかる。

- 1. ~5. のように本来のメッセージの持つ意味毎に適当な一定の言葉を付加する規則を用意しないと、同じ名前のメッセージができてしまう。
- 6. ~8. のように、漢語的表現のメッセージ名で更に似通った名前になり、読み違える可能性が大きい。
- 9. のような名詞で終るものは、動詞がないので正確な意味を読み取ることが難しい。
- 9. のように長いメッセージ名になると意味を正確に理解することが難しくなり、意味の読み取り違いを起こす場合がある。

ATMの仕様記述の中で、上記の問題を含んでいたメッセージが記述単位当りにどのくらい存在していたか調べたものと、記述単位当りの全メッセージ数とそのメッセージに用いられる単語数の比較が以下の表1である。半数近いメッセージに対して効果が得られ、辞書に登録する語の数も1/3以下になることが確認できた。

表1 全メッセージに対する、効果のあるメッセージの割合と辞書登録数の割合

	取引の全メッセージに対する割合
効果のあるメッセージ	44.5 [%]
辞書に登録する語の数	30.2 [%]

4. まとめ

本報告では、形式的仕様記述において計算機が意味解析が可能で、人間にとってわかりやすいネーミング手法について述べた。今回我々が適用したATM向けCASEにおいて、従来の厳密さを維持したまま仕様を読みやすく、意味を正しく理解することができるようになることが確認できた。

今後は、より多くのソフトウェア開発支援を可能にするために、ATM向けCASEという特定分野のCASEから、より汎用的なCASEにおいても適用することができるよう改良を進める。

参考文献

- [1] Jungclaus, R. and Saake, G.: Formal Specification of Object Systems, in TAPSOFT '91, pp.60-82, Berlin(1991), Springer, LNCS 494
- [2] Ehrlich, H. D. and Gogolla, M.: Objects and Their Specification, in 8th Workshop on Abstract DATA Types, pp.40-65, Berlin(1993), Springer, LNCS 655
- [3] 清水他, “金融機器組込みソフト向けCASE”, 情報処理学会 ソフトウェア工学研究会, 93-SE-91, 1993