制限された日本語ユースケース記述からシーケンス図への変換

奥村 和恵† 金澤 典子† 塚本 享治†
東京工科大学メディア学部メディア学科†

1.はじめに

UMLなどの図を中心に開発を行う、モデル駆動型の開発手法がある。しかし図の作成は慣れない開発者にとって難しい。そこで本稿は仕様書の一種であるユースケース記述に含まれるUML図の要素に着目し、シーケンス図とクラス図に変換することを取り上げる。ユースケース記述は自然言語であるため、図の要素が上手く抽出できない。ユースケース記述の文法を限定することで抽出しやすくし、変換ツールを作成して実験を行ったので報告する。

2.ユースケース記述からUML図への変換手順

複雑になったソフトウェア開発を少しでも簡単にするために、オブジェクト指向開発法が登場した。中でもモデル駆動型開発手法では、UMLなどの図を中心に開発する。またICONIX法はユースケースが開発の中心にあり、UML図のクラス図とシーケンス図、ユースケース記述などの相互更新により開発する。

ユースケースの聴取から、ユースケース記述の作成、UML図作成まで、全て手作業で行う。しかし図の作成は難しく、時間と手間が掛かる。特に設計の要となる図は失敗できない。そこで図制作者の技術不足を補い、開発を簡単にするために、図生成の自動化を試み、図 1のような自動変換ツールを作成した。

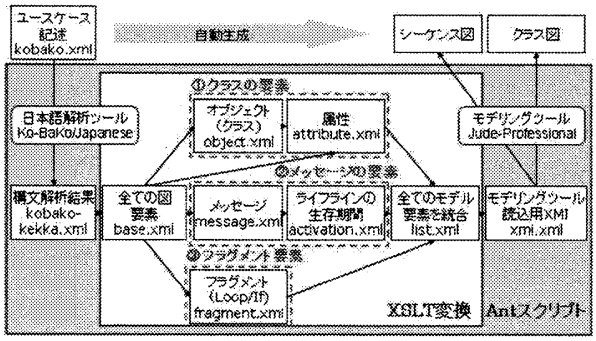

図 1 変換の全体像

日本語で書かれたユースケース記述を用意して、多機能日本語処理ライブラリ[Ko-BaKo/Japanese][1]で構文解析を行った。解析結果から図に必要な要素を抽出してXMI形式へ変換した。最後にUMLモデリングツール[Jude-Professional][3]に読み込ませて、シーケンス図とクラス図を生成した。変換は全てAntスクリプトで行った。

Generating sequence diagrams from limited use case descriptions of grammar
†Kazue Okumura, Noriko Kanazawa, Michiharu Tsukamoto
Tokyo University of Technology, School of Media Science

3.構文解析結果からUML図要素の抽出と変換

3.1. Ko-BaKo /Japaneseによる構文解析

用意した日本語ユースケース記述を構文解析し、解析結果からUML図の要素を集めた。図の要素の判定方法は、単語の接続詞による抽出と、文全体の係り受け構造による分析からである。

ユースケース記述には単文だけではなく複文も登場する。複文は複数の述語を持つ。述語に付く接続詞の解析結果「SOSHITE」が出力されるので、文がどこで分割できるか判定できた。

日本語では重複する主語や目的語は省略される。シーケンス図の送信元・受信先ライフラインにあたり、省略されると図が生成できない。省略された語は文脈分析によって補われる。

3.2. 接続詞によるUML図の要素の抽出

UML図の要素は変換ツールが文中の単語の役割(主語、目的語など)から判断した。単語の役割は変換ツールが接続詞から判断した。各要素の修飾句も、変換ツールが接続詞の解析結果から判断した。図 2のようにユースケース記述の処理文から、シーケンス図とクラス図の要素が抽出できた。

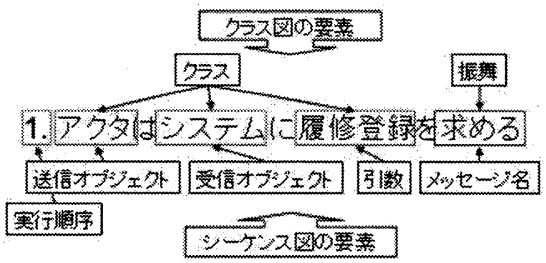

図 2 通常処理文とUML図の要素の対応

3.3. 抽出した要素の精製とUML図の生成

変換ツールはUML図の生成のために、図の要素を抽出し、必要な情報を付加した。抽出して精製したUML図の要素がリスト 1である。

精製した図要素から UML モデリングツールへ読み込ませる XMI ファイルを、変換ツールが生

成した。モデリングツール内でシーケンス図とクラス図を自動生成させた。

リスト 1　抽出した UML 図の要素

```
<XML>
 <object>                                          オブジェクト要素
  <no>1</no>   <name>学生</name>
 </object>・・・・・・
 <attribute class="1" type="int">                  属性要素
  <no>1</no>   <name>学籍番号</name>
 </attribute>・・・・・・
 <message>
  <no>1</no> <step> 1 </step> <name>開始</name>
  <arg type="2">履修登録</arg> <type>create</type>
 </message>・・・・・・                             メッセージ要素
 ・・・他の UML 図要素・・・
</XML>
```

4. 日本語ユースケース記述文法の提案

従来のユースケース記述では正確な解析や変換ができない。そこで UML 図変換専用として、次のようなユースケース記述文法を設けた。

- 内部処理を記述する
- 登場するオブジェクト名を統一する
- ユースケース記述をサブシステム毎に書く

この他にも構文解析や要素の抽出をしやすくするために、細かい文法制限を設けた。

5. 変換実験と考察

5.1. 実験結果

上記の規則に従って変換実験を行った。結果、ユースケース記述から、図 3のシーケンス図と図 4のクラス図が自動生成できた。

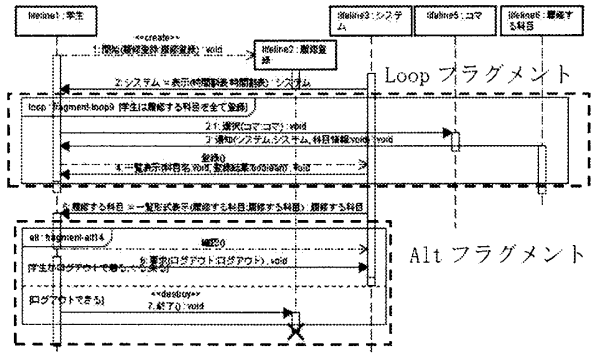

図 3　自動生成されたシーケンス図

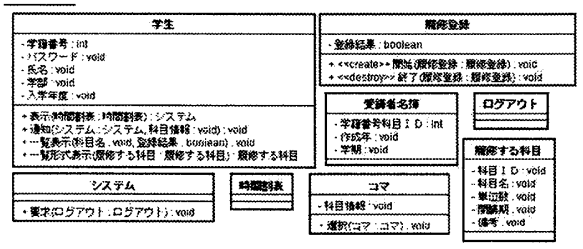

図 4　自動生成されたクラス図

5.2. 成果と問題

変換に成功したのは、変換元のユースケース記述を制限したからである。特に内部処理の記述とオブジェクト名の統一は、より正確な図に変換するのに大きな効果があった。内部処理の記述により、インターフェイスしかないシーケンス図ではなくなった。オブジェクト名の統一は同じクラスが重複して生成されるのを防いだ。

シーケンス図で繰り返しや条件分岐を表す時にフラグメントを使用する。[1]の研究ではフラグメントを再現できなかった。変換ツールではメッセージの位置からフラグメントの開始位置を計算ことにより、正確なフラグメントが生成できた。

しかしクラスの関連性は表現できなかった。ユースケース記述に全く情報が載っていないからである。

5.3. ユースケース記述のパタンとひな型の適用

ユースケース記述はユースケースレベルと、記述内の処理文レベルの 2 段階で分類できる。

ユースケースレベルでは、ログインとログアウト、検索の 2 種のパタンが見つかった。また CRUD(生成, 検索, 更新, 削除)という 4 種へ分類できた。しかし UML 図に変換する際、種類に応じたひな型を適用する問題が残されている。

ユースケース記述内の処理文は、起動処理、終了処理、通常処理の 3 種に分類できた。起動処理と終了処理は日本語文章に特徴が見られた。シーケンス図に変換する際、この特徴からメッセージの種類 (Create, Destroy) を判別し、より詳しいシーケンス図が作成できた。

6. おわりに

実際に実験ツールを作り、ユースケース記述から UML 図を自動生成した。ユースケース記述に文法制限を設けることで、より正確な図の生成ができた。また UML 図へ変換する際に一定のひな型を適用できるものと思われる。

参考文献

[1] 広瀬希美, “ユースケース記述から UML 図の生成”, 情報処理学会 第 69 回全国大会 4M-2, 2006

[2] “Ko-BaKo/Japanese”, ㈱日本システムアプリケーション, http://www.jsa.co.jp/LANG/ko-bako/index_frame.htm

[3] “Jude Professional”, ㈱チェンジビジョン, http://jude.change-vision.com/jude-web/index.html