4M-2

ユースケース記述から UML 図の生成

広瀬　希美†　　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

現在のソフトウェア開発では、オブジェクト指向開発法が広く採用されている。その方法は、要求定義・分析設計・実装というプロセスで進む。分析フェーズで非常に重要な役割を果たすのがユースケースである。ユースケースはユースケース図とユースケース記述で構成される。本稿では日本語で書かれたユースケース記述からシーケンス図を生成するツールを作成したので報告する。

2. UML 生成システムの全体像

UML はその活用にオブジェクト指向を理解しダイアグラム作成には知識と経験が必要である。しかし、ユースケース記述は自然言語を用いるためだれでもが書けそうに思えるが論理的に正しい記述となるとそれほど簡単ではない。ユーザの言葉で定義したユースケース記述を解析し、オブジェクト指向の知識を補うことによりシーケンス図を生成し、シーケンス図からソースコードの生成を行うシステムを試作した。その手順を図 1 に示す。

ここでは小規模な開発を想定する。小規模なオブジェクト指向開発で使われる ICONIX プロセスでは、要求定義を示すユースケース記述、静的な部分を表すクラス図、動的な部分を表すシーケンス図によってソフトウェアの設計を行う。通常、開発者自身がユースケース記述をもとにシーケンス図を作成する。本システムでは図 1 の方法でソースコードを生成する。

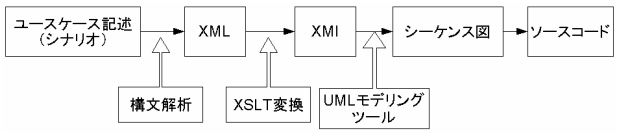

図 1：UML 図生成プロセス

まず、シナリオを多機能日本語処理ライブラリ Ko-BaKo[1]を用いて解析し、シーケンス図を作成するために必要な情報を XML 形式で抽出した。これを XSLT 変換してシーケンス図に対応する XMI を作成した。最後に、モデリングツール・Enterprise Architect[2]を利用し XMI を読み込むことでシーケンス図を生成した。

3. シナリオの構文解析

3.1. シナリオ構文の定義

シナリオは「ユースケースを実現するためのシステムのふるまいを表す文章」である。その記述構文として図 2 のような助詞を使う構文を定義し、シナリオはこの構文で書くこととした。

管理者 [が] 操作画面 [に] 入力情報 [を] 入力する

図 2：シナリオ構文

この構文を書く上で定めた記述方法を以下に述べる。

- 文章ごとに構文解析を行う。
  - 句・節は取り扱わない。
- シーケンス図に対応する文章の形式
  - 1 文ごとに番号をつける。
  - 特殊要素は次のように決めた。

(1) メッセージ情報
生成メッセージ:述語に必ず「作る」と記述する。
停止:述語に必ず「削除する」と記述する。
(2) 複合フラグメント
繰り返し:「2 から 3 を繰り返す」と繰り返したい文章の番号を指定する。
条件分岐:「もし「条件」なら」と構文の前に記述する。

3.2. 情報の抽出

シナリオから情報を抽出するには構文解析ツールを利用した。助詞によってシーケンス図と対応する単語を抽出し、XML 形式で表した。図 3 のようにシナリオ構文とシーケンス図が対応している。

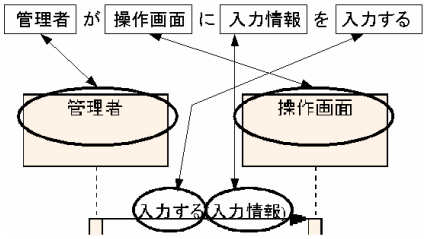

図 3：シナリオとシーケンス図の対応関係

図 3 より、助詞と情報との対応関係は次のようなる。
「が」：メッセージを送るオブジェクト名
「に」：メッセージを受取るオブジェクト名
「を」：引数
述語 ：メッセージ名

これらの情報をもとに XMI を生成することによりシーケンス図が作成されると考えた。

4. UML モデリング入力形式への変換

4.1. シーケンス図情報の変換

シナリオから抽出した情報を整形し、id などの必要情報を加えることによって XML ファイルを作成した。これがシーケンス図に対応する XMI の生成に必要な情報一覧

Generating UML diagrams from use case descriptions
†Nozomi HIROSE, Noriko KANAZAWA, Michiharu TUKAMOTO
Tokyo University of Technology, School of Media Science

となる。
(1) メッセージ情報
1 つの構文が 1 つのメッセージ情報となっている。構文解析結果ごとに情報を取り出し、タグ付けを行った。メッセージを送るオブジェクトを sender、受取るオブジェクトを receiver とした。
(2) オブジェクト情報
(1)で生成したメッセージ情報から必要な情報を抽出した。sender、receiver はオブジェクト名なので、重複を取り除くことでオブジェクト一覧とした。
(1)(2)で生成した情報を図 4 のような XML 形式にして、次の「XMI の生成」の入力とした。また、オブジェクト、メッセージをそれぞれ識別するために id を導入した。XMI では、sender と receiver を id で指定するため、オブジェクト名ではなく対応するオブジェクト id とした。

```
<?xml version="1.0" encoding="Shift_JIS"?>
<一覧>
    <オブジェクト xmlns:set="http://exslt.org/sets">
        <name>管理者</name>
        <id>ob1</id>
        <no>1</no>
    </オブジェクト>
    ・・・
    <メッセージ>
        <name>入力する</name>
        <id>m1</id>
        <引数>ID</引数>
        <sender_id no="1">ob1</sender_id>
        <receiver_id no="2">ob2</receiver_id>
    </メッセージ>
    ・・・
</一覧>
```

図 4：シーケンス図情報

4.2. XMI の生成

XMI(XML Metadata Interchange)は、UML などメタデータ交換のための標準フォーマットである。本研究では、シーケンス図を生成するために最低限必要な情報のみを扱った。また、XMI で扱う id には規定があり、英字のみまたは英字と数字が混ざったものが使用できる。

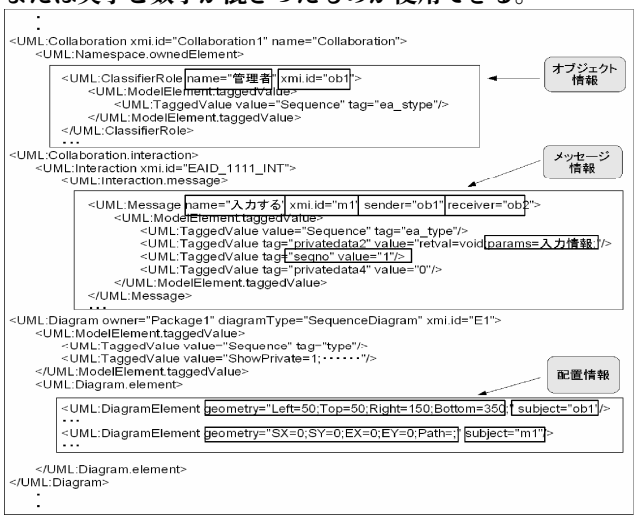

図 5：XMI

(1) オブジェクト情報・メッセージ情報
4.1 で抽出した情報をそれぞれ必要な箇所に組み込んだ。メッセージ情報では、メッセージの順を設定した。
(2) 配置情報
配置箇所とどのオブジェクト、メッセージに対応しているかを id で設定する。オブジェクトの配置は、座標を 4 ヶ所指定する。これがオブジェクトの上下左右の値になる。メッセージの配置は、4 ヶ所とも 0 を指定する。メッセージ情報で設定した seqno の順に表示される。

5. 実験と考察

本システムを利用して、実際にいくつか XMI を生成し、ツールに読み込んでシーケンス図を生成した。ほぼ予想と同じシーケンス図を生成することができた。(図 6)

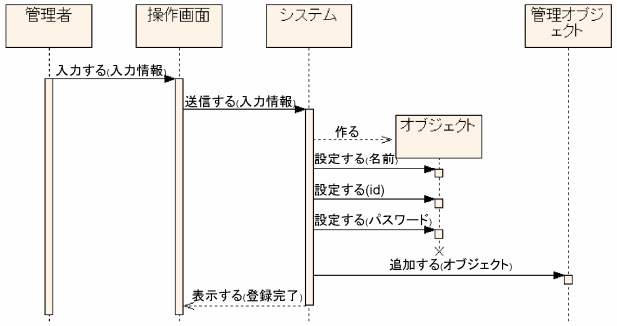

図 6：生成されたシーケンス図

しかし、次のような問題も残っている。
(1) 構文解析
シナリオの構文解析では、句・節まで扱わず、単語に注目した。句・節まで考慮することによりさらに本格的なシーケンス図を作成できると考える。
(2) 複合フラグメント
XMI での配置は数値を指定することで行っている。しかし、複合フラグメントはどのメッセージに対応するのかが重要である。メッセージ id で配置を指定できるようになれば複合フラグメントを自動的に配置できると考える。
(3) シナリオ入力画面の注意事項
シナリオの入力画面に、注意事項を書く必要がある。これは、構文とちがう文章を書いた場合に誤った情報抽出が行われるためである。また記述が決められている特殊要素で同じ意味でもちがう記述をした場合、特殊要素と判別されない。

6. おわりに

本システムはシナリオの書き方を定義し、構文解析することによってシーケンス図に必要な情報を取得し、XMI を生成するものである。ツールを利用し生成された XMI を読込むことでシーケンス図を生成することができた。
現段階では、操作を手動で行わなければならないことや、より細かい構文解析、複合フラグメントの配置方法など課題が残っている。これらの課題を解決することにより、より本格的なシーケンス図の自動生成ができると考えられる。

参考文献

[1] (株)日本システムアプリケーション, 多機能日本語処理ライブラリ Ko-BaKo/J
[2] (株)スパークスシステムズジャパン, “Enterprise Architect”, http://www.sparxystems.jp/