F-028

# VoiceXML と UML を用いたオブジェクト指向的対話システム開発手法の提案

## Object-oriented Dialogue System Development method using VoiceXML and UML

荒木雅弘
Masahiro Araki

## １．はじめに

近年の音声認識・合成技術の進歩を背景として、音声対話システムの実用化への期待が高まっている。WWW の標準化団体である W3C においては Voice Browser Working Group が活動を行っており、VoiceXML2.0 の仕様が固まりつつある[1]。このような状況にも関わらず、ボイスポータルや観光情報提供のシステムなどは試験サービスの域に留まっており、音声対話システムは一般に広く普及しているとはいえない状況である。

音声対話システムが普及しない原因として、インタフェース技術としての未熟さが挙げられることが多い。しかし、我々はこの問題を音声対話システムを作成する際の方法論不在の問題として捉える。音声対話システムの開発には、文法記述・状態遷移記述・アプリケーションインタフェースの利用など異なった分野のノウハウが必要であり、このことが開発コストを押し上げ、普及を阻害する要因となっていると考えられる。このノウハウを開発方法論としてまとめることで、音声対話システムの開発コストが下がり、同時に信頼性・保守性を高めることによって、その普及に寄与すると期待できる。

現在、ある程度以上の規模のソフトウェア開発にはオブジェクト指向開発方法論が用いられることが多く、その効果が多くの事例で確認されている。オブジェクト指向設計・開発は、ソフトウェアの部品化、パターンの共有、開発効率の向上、信頼性・保守性の向上が主たる特徴であり、音声対話システムの開発においても、アプリケーションロジック部分の作成においては適用可能な方法論である。しかし、音声対話部分の中心となる対話記述言語 VoiceXML は、対話パターンの記述に手続型プログラミング言語の制御構造を混在させた形式を持ち、オブジェクト指向開発との相性は良いものとは言えない。

そこで我々は、オブジェクト指向設計に用いられる UML (Unified Modeling Language) をモデルとし、そのモデルから VoiceXML をコードとして生成するモデル駆動アーキテクチャを提案する。全体の設計や部品の再利用を UML で行い、その設計結果を VoiceXML コードとして出力することで、VoiceXML の言語仕様にオブジェクト指向的な拡張を加えることなく、対話システム全体としてオブジェクト指向開発が行えることになる。

## ２．UML を用いた対話システムの設計

UML はソフトウェアシステムの静的・動的構造をモデル化するもので、目的に応じて 9 種類のダイアグラムが定義されている。音声対話システムにおいては、個々の対話部品がどのような機能を持つかという情報と、どのように対話が遷移するかという情報が重要である。従って、本提案で採用する UML は、クラスの静的構造を表現するクラス図と、オブジェクトの相互作用を表現するコラボレーション図とする。

### 2.1 クラス図による対話の記述

個々の VoiceXML ファイルは UML のクラス図におけるクラスに対応付ける。例としてチケット予約の音声対話システムを想定し、ユーザ情報を収集する部分の対話を考える。対話 slot1 はユーザの氏名・電話番号・希望の公演日・枚数を取得する副対話からなる。この対話は予めドメイン独立で用意されているクラス slotFilling のサブクラスとして実現される(図 1)。

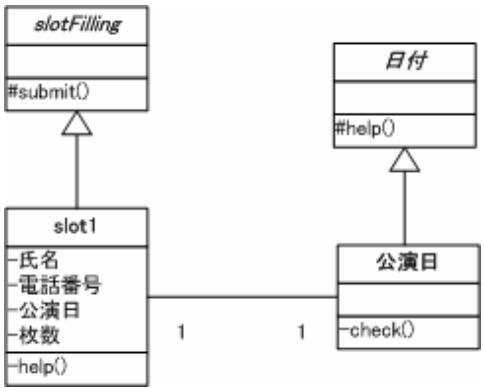


図 1 クラス図による対話部品の表現

スーパークラス slotFilling は、対話によって埋めるべき変数の入れ物として機能し、変数の値をサーバへ送信する部分(submit)のみをメソッドとして持つ。具体的に対話によって埋める変数とその型はそのサブクラスである slot1 で定める。

サブクラスの変数フィールドには氏名・電話番号・公演日・枚数の記述があるが、これをオブジェクト型として考える。氏名や電話番号などは、対話によって値を取得するクラス(プロンプトと文法からなる副対話)に対応する。また、公演日のように、単純な日付型のクラスを用いたのでは問題が生じる場合(例えばチケット販売のない日は指定できないようにしなければならない)、日付クラスを継承した公演日クラスを作成し、メソッド check によって公演設定のない日が指定された場合の処理を行う。

### 2.2 コラボレーション図による状態遷移の記述

音声対話システムは、情報の流れる方向に基づいて、ス

京都工芸繊維大学, Kyoto Institute of Technology

ロットフィリング、データベース検索、説明の 3 種類のタスクに分類することができる[2]。提案手法では、それぞれのタスクに関して、典型的な対話状態遷移をコラボレーション図で記述しておき、それぞれのドメインに応じてカスタマイズすることとする。コラボレーション図に出現するオブジェクトは、典型的な対話クラスのオブジェクトであるとする。

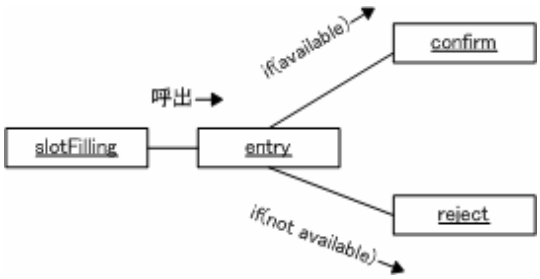


図 2 コラボレーション図によるスロットフィリングタスクの表現

図 2 に出現する slotFilling, confirm, reject は抽象クラスであり、クラス図で具象クラスを定義することによって、実際の VoiceXML ファイルを生成できる。図 2 中の entry はサーバ側のプログラムで、slotFilling によって得られた値でチケット予約可能かどうかの判定を行い、その結果によって confirm または reject のいずれかに進む。この部分はスロットフィリングのドメインが変更されても動作に違いはないと考えられるので、クラス図で具象クラスを定義する必要はない。

## ３． UML からの VoiceXML 生成

### 3.1 状態に対応する VoiceXML の生成

スロットフィリング、データベース検索、説明に対応するライブラリは、それぞれのタスクの典型的な状態とその遷移情報から構成されている[3]。クラス図では、典型的な状態を表す抽象クラス(slotFilling, confirm など)を継承し、具象クラス(slot1 など)を作成する。クラス図上は、slot1 は slotFilling クラスを継承しているが、VoiceXML には継承の機能がないため、抽象クラスの要素をコピーして具象クラスを実現する。

例えば、slot1 は図 3 に示すような VoiceXML ファイルとなる。

```
<?xml version="1.0" encoding=”Shift_JIS” ?>
<vxml version="1.0">
   <form>
      <subdialog name="氏名" src="name.vxml" />
      <subdialog name="電話番号" src="tel.vxml" />
      <subdialog name="公演日" src="公演日.vxml" />
      <subdialog name="枚数" src="count.vxml" />
      <filled>  <submit next="entry.jsp"/>  </filled>
    </form>
</vxml>
```

図 3 VoiceXML によるスロットフィリング対話

### 3.2 副対話に対応する VoiceXML の生成

図 3 の subdialog で指定された VoiceXML ファイルは、氏名や金額などを入力するための副対話としてライブラリ化されているものである。具体的には、VoiceXML の言語仕様で定義されているもの(boolean, date, currency など)は、filed 要素にプロンプトを追加して個別の VoiceXML ファイルとして用意する。

一方、クラス図で拡張された公演日は、日付型のライブラリをコピーし、その filled 要素に check メソッドを記述するコメントを埋め込んで VoiceXML ファイルを生成する。メソッドは if 要素などの制御構造を組み合わせて記述するか、ECMAScript として記述する。

### 3.3 アプリケーションロジックの生成

図 2 の entry オブジェクトは、slotFilling 対話によって得られた変数値を使ってデータベースにアクセスし、チケットが予約可能であるかどうかを判定するアプリケーションロジックである。ここでは、VoiceXML との整合性を考慮し、JSP (JavaServerPages)で記述する。テンプレートの entry オブジェクトは全ての変数値の AND 条件でデータベース検索を行う JSTL(JSP Standard Template Library)が設定してあり、各ドメインに応じて条件を変更し、検索結果を受けた分岐を行う部分などを設計者が記述する。

## ４． おわりに

本稿では、VoiceXML と UML を用いてオブジェクト指向設計の枠組みで音声対話システムを実装する方法論を提案した。オブジェクト指向設計の部分を UML に限定し、そこから対応する VoiceXML を生成する方法を取ることによって、VoiceXML に拡張を施す必要なしにオブジェクト指向的な開発が行える。

関連するアイデアとしては、対話の状態遷移を可視化するツール[4]が存在する。しかし、このツールはコーディングを視覚的に行うためのもので、設計方法論を提案するものではない。

今後は、データベース検索・説明タスクにおけるライブラリの実装を試み、本提案の妥当性を検証する予定である。

## 参考文献

[1] Scott McGlashan, *et.al.*：Voice Extensible Markup Language (VoiceXML) version 2.0, http://www.w3.org/TR/voicexml20/ 2003.

[2] Masahiro ARAKI, Tasuku ONO, Kiyoshi UEDA, Takuya NISHIMOTO, and Yasuhisa NIIMI: ”An Automatic Dialogue System Generator from the Internet Information Contents”, Proc, EUROSPEECH2001, pp1150-1156, 2001.

[3] 荒木 雅弘，平田 大志，駒谷 和範，堂下 修司: 音声対話システム構築のための対話ライブラリ , 人工知能学会研究会資料, Vol. SIG-SLUD-9901-1, pp. 1-6, 1999.

[4] 株式会社プロトン: VBVoice 4.4, http://www.sb.proton.co.jp/product/vbvoice/vbvoice2.html.