オブジェクトの継承関係を利用したスマートスピーカーの応答定義

本多 佑希[1]　岸本 有生[2]　兼宗 進[1]

概要：スマートスピーカーは音声の入力を解釈し，音声で応答を返す．その際，鳥のような上位の概念についての定義を行えば，スズメのような下位の概念を個別に定義する必要をなくすことができると考えた．音声の応答定義をオブジェクト指向言語のプログラムとして記述することで，語彙の上限関係をオブジェクトの継承で表現し，鳥という上位語で飛ぶことが定義されていれば，スズメのような下位語ではポリモルフィズムにより個別の定義を不要にすることができる．本発表では，教育用プログラミング言語「ドリトル」と，LINE 社のスマートスピーカーである「Clova」を用いて構築した学習システムを紹介し，工学系の大学生を対象に行った授業について報告する．

キーワード：スマートスピーカー，プログラミング教育，オブジェクト指向

A Response Definition Method for Smart Speaker using Inheritance of Object Oriented.

Abstract: By defining a correspondence between an input sentence and a response sentence, a smart speaker can interpret the voice input and return a voice response. However, the possibility of a variety of words being input into the smart speaker has made the definition complicated. In this paper, we propose a method for defining the response to a voice response that allows us to omit the definition of lower-level concepts such as sparrow, if we define higher-level concepts such as bird. By writing the response definitions of speech as an object-oriented language program, the hierarchical relationship of the vocabulary is represented by object inheritance. If flying is defined in the upper class of birds, the definition can be omitted in the lower class, such as sparrows, by polymorphism. In this presentation, we introduce a learning system built using the educational programming language "Dolittle" and LINE's smart speaker "Clova". We also report on a class we conducted for undergraduate engineering students.

1. はじめに

スマートスピーカーのアプリケーション開発を支援するプログラミング学習環境の開発を進めている．スマートスピーカーはユーザが音声によってアプリケーションの起動や操作を行うデバイスである．「10 時に起こして」といった文章から「10 時」や「起こして」といった語句を AI を用いた音声認識により抽出し，適切な処理を行ってユーザに応答する．

スマートスピーカーに認識させるテキストを「すずめは飛ぶ？」「ペンギンは飛ぶ？」といった文章などの「名詞」と「動詞」がセットになった文章に限定することによって，名詞をオブジェクト，動詞を命令として扱うことができると考えた．

すずめやペンギンをオブジェクトとして扱うにあたり，それらは鳥に含まれるといった語彙の階層構造に着目した．鳥をオブジェクトとして定義し，「飛ぶ？」という命令を定義した場合には，鳥を継承したすずめやペンギンオブジェクトはポリモルフィズムにより「飛ぶ？」を定義しなくても参照することができる．日常的に用いる語彙階層の概念をオブジェクト階層の形に落とし込む学習であれば，初学者でもオブジェクトの階層構造を想像しながら設計することや，オブジェクトの階層構造を直感的に想像することが可能なのではないかと考えた．

[1] 大阪電気通信大学
Osaka Electoro-Communication University
[2] 大阪電気通信大学高等学校
Osaka Electoro-Communication University High School

また，教育用プログラミング言語ドリトルではプロトタイプ方式のオブジェクト指向を採用していることから，プログラミングを専門的に学習したことがない中学生・高校生でもオブジェクト指向を活用してプログラミングできることが知られている．そのため，考案した学習手法をドリトル言語を使ったプログラムで実習することにより，初学者でもオブジェクトの継承などの概念を理解することができるのではないかと考えた．

本研究は，中学生・高校生を対象としてオブジェクトの階層構造の概念を理解させることが目標である．題材としてスマートスピーカーのアプリケーション開発を採用していることから，高等学校での情報システムや中学校の双方向コンテンツの学習に活用することができる．また，鳥やすずめなどの関係をオブジェクト指向の考え方を流用してプログラムで表現する活動は，高等学校で重要視されているモデル化に近いと考えている．これらの学習をメインとして授業を展開しつつ，オブジェクトの階層構造の概念を自然と身に付けさせることができると期待している．

今回は予備実験として，検討した記述方式を用いて大学生を対象に授業を行った．ルールの妥当性を確認するとともに，記述方式を応用して，学生が自分でイメージしたモデルをプログラムに落とし込むことができるかを確認した．本稿では記述方式を提案するとともに，授業の結果について報告する．

2. スマートスピーカー

スマートスピーカーはユーザが音声によりアプリケーションの起動や操作を行う機器である．ユーザが発した音声をサーバに送信し，サーバが音声認識エンジン (AI) を用いてテキストに変換し，適切な処理を行う．システムを簡略化した概要図を図 1 に示す．

3. 記述方式

スマートスピーカーには多種多様なアプリケーションが存在する．それぞれのアプリごとに起動・操作のためのテキストは異なる．しかし，以下のような形の短文を認識できれば基本的なスマートスピーカーの操作を表現することができるのではないかと考えた．

(1) A の B は？
(2) A は B？
(3) A を B

これらの文章をプログラム上で表現するために，図 2 の記述方式を用いる．

これらの文章は全て A が名詞，B が動詞に対応している．A をオブジェクト名，B を命令名とすることで，文章をオブジェクト指向のルールで表現することができる．

「すずめは飛ぶ？」という文章を認識した際の動作は，「すずめ」オブジェクトに「飛ぶ？」という命令として定義する．これにより，「A は B？」という文章を表現する．

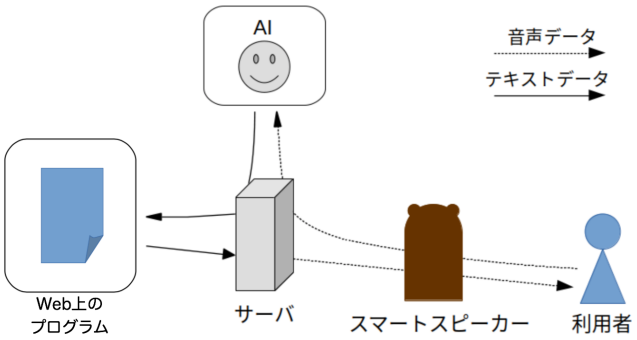

図 1 一般的なスマートスピーカーのシステムモデル
Fig. 1 A system model of the smart speaker.

「A の B は？」という文章も，例えば「今日の天気は？」をプログラム上で表現する際には，「今日」オブジェクトに「天気？」という命令として定義する．「天気」はあくまで名詞であるが，「？」をつけることによって，天気を教えてほしいという動詞の形を省略する形で表現している．

「A を B」という文章も，同様に表現できる．例えば，「夕食を考えて」という夕食提案アプリを考えた際には，「夕食」オブジェクトに「考えて」という命令を定義することで表現する．

4. オブジェクト階層を用いた語彙階層の表現

4.1 問題

3 章で提案した記述方式によって，短文を認識する．プログラムの構造や，使用する機能についてはオブジェクトの生成と命令定義しか行っていないため，初学者でも十分に理解できると考えられる．しかし，図 3 のようにすずめやハト，トキなど複数の鳥を認識するプログラムを記述した際には，同じような命令定義が複数回登場し，再利用性が考えられていないことがわかる．

4.2 検討

例に挙げているすずめやハト，トキやツバメなどは「鳥」という分類に含まれる．今回は，こういった「語彙の階層構造」に着目した．語彙の階層構造としては，広く使われているものとしてシソーラスが存在する [3]．シソーラスで扱われる上位語や下位語，広義語や狭義語といった考え方を参考にしながら簡易化することで，単語同士の関係をプログラム上で表現できるのではないかと考えた．この階層構造を図で表現したものを図 4 に示す．すずめやハトなどの単語が，鳥から派生した形になっている．

すずめやハトなどの単語が鳥から派生している構造を考えた際に，プログラミング言語の「オブジェクトの階層構造」に類似していることに気付いた．「鳥」というオブジェクトを「すずめ」や「ハト」などのオブジェクトが継承している．この「語彙の階層構造とオブジェクトの階層構造

```
1 すずめ＝単語！作る.
2 すずめ:飛ぶ？＝「クローバー！"飛びます．"話す.」.
3 今日＝単語！作る.
4 今日:天気？＝「クローバー！"雨が降るでしょう．"話す.」.
5 夕食＝単語！作る.
6 夕食:考えて＝「クローバー！"カレーはいかが？"話す.」.
```

図 2 短文を認識した際の処理を定義するプログラム
Fig. 2 A program for recognizing short sentence.

の類似点」を利用することで，語彙の階層構造をプログラムとして表現することができると考えた.

4.3 オブジェクトの階層構造

今回対象としているプログラミング言語「ドリトル」はSelf で知られるプロトライプ方式のオブジェクト指向を採用している．C++などと違いクラスという概念はなく，オブジェクト同士がプログラム実行中に動的に継承関係を作る.

今回の場合，図 4 の「鳥」「すずめ」はそれぞれがオブジェクトであり，クラスとオブジェクトという関係ではない．そのため，今回のこの関係を図示する際には，オブジェクト図は不適切である．イメージではクラスベースの基底クラス，派生クラスという関係が近い．そのため，これらの関係を図示する場合にはクラス図の方が適切であると考える．したがって本稿ではこれらのオブジェクトの関係を示すためにクラス図を用いるが，厳密なものではなく，関係を示すためのものと認識されたい.

4.4 表現方法について

4.2 章で検討したように，今回は語彙の階層構造をオブジェクトの階層構造で表現することにした．図 5 に，図 3 を 4.2 章で検討した方式で表現し直したプログラムを示す．親オブジェクトである「鳥」のメソッドとして「飛ぶ」を定義しており，子オブジェクトである「すずめ」「ハト」などは継承関係を辿って「飛ぶ」を参照することができる．この方式により，再利用性が向上されたプログラムを実現することができた.

4.5 学習活動を通して得ることができる知識

本節で検討，提案した「語彙の階層構造をオブジェクトの階層構造で表現する手法」を用いることで，オブジェクトの階層構造（継承関係）を初学者でも体験的に学習することができる．例えば，「鳥」という言葉を見たときには，その中には「すずめ」「ハト」なども含んだものだと自然と理解することができる．「近畿地方の天気予報」を見たときには，その中には兵庫県や大阪府などが含まれていると自然と理解することができる．そういった「語彙の階層構造」をプログラムから表現する学習活動は，オブジェクトの階層構造の学習に適していると考えている.

```
1  すずめ＝単語！作る.
2  すずめ:飛ぶ？＝「
3    クローバー！"すずめは飛びます．" 話す.
4  」.
5  ハト＝単語！作る.
6  ハト:飛ぶ？＝「
7    クローバー！"ハトは飛びます．" 話す.
8  」.
9  トキ＝単語！作る.
10 トキ:飛ぶ？＝「
11   クローバー！"トキは飛びます．" 話す.
12 」.
13 ツバメ＝単語！作る.
14 ツバメ:飛ぶ？＝「
15   クローバー！"ツバメは飛びます．" 話す.
16 」.
```

図 3 図 2 の「すずめ」を様々な種類の鳥に適用したパターン
Fig. 3 A program of some bird definitions.

```
鳥
 ├ すずめ
 ├ ハト
 ├ トキ
 └ ツバメ
```

図 4 語彙の階層構造
Fig. 4 A word hierarchy structure of birds.

```
1 鳥＝単語！作る.
2 鳥:飛ぶ？＝「|A|
3   クローバー！（A+"は飛びます．"）話す.
4 」.
5 すずめ＝鳥！作る.
6 ハト＝鳥！作る.
7 トキ＝鳥！作る.
8 ツバメ＝鳥！作る.
```

図 5 図 3 を語彙階層とオブジェクト階層の考え方を使って表現し直したプログラム
Fig. 5 A program using the word hierarchy structure.

今回ベースの言語として採用しているドリトル言語はオブジェクト指向の理解を目的の一つとしている．兼宗ら [6] は，プロトタイプ方式のオブジェクト指向を採用した利点として，『クラス定義が不要である.』『複雑なクラス階層を理解する必要がない.』『複製というオブジェクトの生成モデルは単純で理解しやすい.』といった点を挙げている．クラスベースのオブジェクト指向言語では本稿で提案した語彙階層を表現するプログラミング学習は現実的ではないため，これらのドリトル言語の利点が大きく寄与している.

一方で，今回提案している語彙階層をオブジェクト階層で表現する方法でスマートスピーカーのアプリケーションを開発する学習は，オブジェクトの階層構造の学習という点において優れているのではないかと考えている.

図 5 で示したプログラム中のオブジェクトの階層構造を図 6 に示す．この例では，ある一つのオブジェクトを親に

持つ複数の子オブジェクトを生成している．この構造は，ドリトルではペイントツールのプログラミングでの「ボタンオブジェクトから GUI オブジェクトを複数生成する」や，タートルグラフィックスの学習の「タートルオブジェクトから複数のタートルを生成する」などの例で用いられる．今回の提案内容による学習では「親オブジェクトにはどのような命令・プロパティを用意するべきか？」といったことも学習者が考えるため，オブジェクトの階層構造をイメージしながら，オブジェクト設計まで扱っていることになる．今回の題材は，オブジェクト指向（特にオブジェクトの階層構造）の学習という視点から見た場合においては，オブジェクト設計まで題材と出来る点で優れているのではないかと考えている．

例に挙げている鳥の生体を教えてくれる鳥図鑑のアプリケーションを題材に考えると，図 7 のようなオブジェクトの階層構造も扱うことができる．鳥からフクロウとハトがそれぞれ生成され，フクロウ，ハトからもそれぞれ子オブジェクトが生成されている．このような深い階層構造も，直感的に扱うことができるのではないかと考えている．

オブジェクトの設計やオブジェクト指向の考え方の学習は初学者には敷居が高く，高校の授業では取り入れられていない．情報処理学会のカリキュラム標準 J17 [4] では，ソフトウェアエンジニアリング領域の中では「ソフトウェア構築」という講義形式の授業の第４回目で扱われている．また，コンピュータ科学領域の中では「プログラミング言語」というエリアの「オブジェクト指向プログラミング」ユニットの Tier1 のトピックスとしてオブジェクト指向設計やクラスが挙げられている他，「情報管理」というエリアの「データモデル」ユニットの Tier2 の中でオブジェクト指向モデルが挙げられている．

オブジェクト指向の技術や，クラス設計などの学習を目的とした研究としても多く扱われている．高井ら [5] は大学３年生を対象とした講義の中で，提示されたクラス図を基に Java のプログラムを記述する課題を出すなどの学習を行っている．

これらのことから，オブジェクトの継承による階層構造は大学生以上を対象とした授業で扱う高度な概念であり，高校生や中学生などを含めた初学者が扱うには難しいことがわかる．そのため，今回提案する手法はプログラミング教育において重要な意味を持つと考えている．

5. 授業実践

5.1 概要

3 章の記述方式，及び 4 章で検討した階層構造のルールの妥当性を検証するために，工学部の大学３年生１１名を対象に実験的な授業を行った．対象の学生はドリトルの講習を事前に受けており，ドリトル言語の構文については熟知している．また，C や Python などの知識もあるため，プログラミングについてはある程度習熟している．

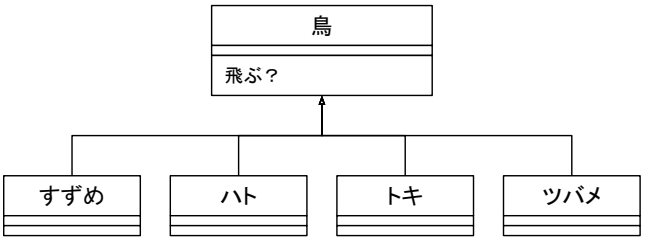

図 6 図 5 のオブジェクトを図にしたもの
Fig. 6 A class hierarcy diagram of birds.

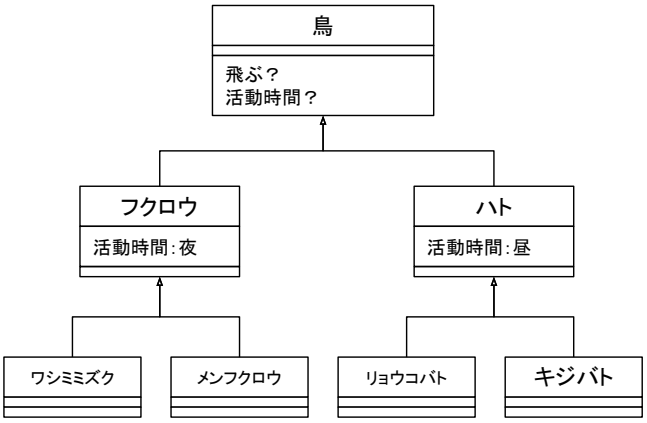

図 7 深いオブジェクトの階層構造の図
Fig. 7 A class diagram with several hierarcys.

5.2 授業内容

対象の学生には，基本的な記述方式のレクチャーを行ったあと，「簡単な演習と，演習の内容を踏まえた課題を解く」という活動を２セット行ってもらった．

１つ目の演習で用いたプログラムを図 8 に示す．「鳥」という基底のオブジェクトに「飛ぶ？」という命令を定義し，殆どの子オブジェクトは継承元の「鳥」の命令を参照するが，「ペンギン」は例外として「飛ぶ？」を上書きする例である．この構造を図示したものを図 9 に示す．

２つ目の演習で用いたプログラムを図 10 に示す．「動物」という基底のオブジェクトに「鳴き声？」という命令を定義し，「鳴き声？」の中では「自分：声」を参照している．「犬」と「猫」がそれぞれ子オブジェクトとして作られ，それぞれに「声」というプロパティが定義されている．この構造を図示したものを図 11 に示す．

また，自分でイメージしたモデルを自分でプログラムの形に組み上げられるのかを測ることでルールの妥当性については検証できると考えたため， それぞれの演習のあとに「演習のプログラムの形で表現できるモデルを自分でイメージしてプログラムすること」という課題を出した．

6. 学生が作ったプログラム

図 12 に演習 1 に対する課題について，学生が書いたプ

```
1  鳥＝単語！作る.
2  鳥:飛ぶ？＝「|A|
3    クローバー！（A+"は飛びます．"）話す.
4  」.
5  すずめ＝鳥！作る.
6  ハト＝鳥！作る.
7  トキ＝鳥！作る.
8  ペンギン＝鳥！作る.
9  ペンギン:飛ぶ？＝「
10   クローバー！"ペンギンは飛ばない"話す.
11 」.
```

図 8 授業内で使用した演習 1 の内容
Fig. 8 A sample program 1 in the lecture.

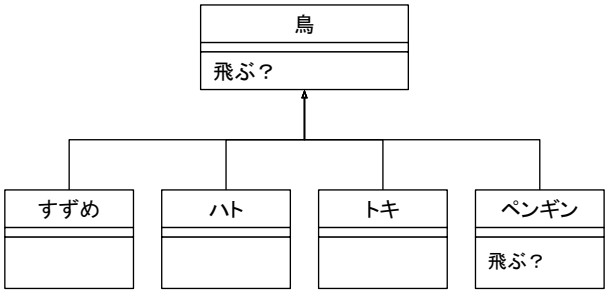

図 9 図 8 の階層構造
Fig. 9 A class diagram of exercize 1.

```
1  動物＝単語！作る.
2  動物:鳴き声？＝「|A|
3    クローバー！（A+"の鳴き声は"+自分:声+です．"）話す.
4  」.
5  犬＝動物！作る.
6  犬:声＝"ワン".
7  猫＝動物！作る.
8  猫:声＝"ニャ〜".
```

図 10 授業内で使用した演習 2 の内容
Fig. 10 A sample program 2 in the lecture.

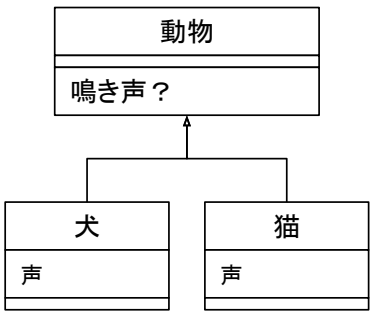

図 11 図 10 の階層構造
Fig. 11 A class diagram of exercize 2.

ログラム例を示す．果物は基本的には甘いという考えのもと，「果物」という既定オブジェクトに「甘い？」命令を追加し，例外である「レモン」は「甘い？」を上書きしている．

図 13 も同様に，OS は基本的には Linux であるという考えのもと，「OS」という既定オブジェクトに「Linux?」命令を追加し，例外である「Windows」「mac」では「Linux?」を上書きしている．OS は基本的には Linux であるという考えが正しいかは別として，演習内容で得た記述方式やオブジェクトの階層関係を自分で理解し，工夫できていることはわかった．

図 14 に演習 2 に対する課題について，学生が書いたプログラム例を示す．「動物」という基底オブジェクトに「場所？」を定義し，実際の住処の情報はそれぞれ子オブジェクトのプロパティに定義している．図 15 も同様に，「物質」という基底オブジェクトに「材質は？」を定義し，実際の材質の情報はそれぞれ子オブジェクトのプロパティに定義している．

今回検討したルールを基に学生が自分で考えてプログラムの形に落とし込めているところから，学生はルールを理解し，自分で工夫できていることがわかる．そのため，今回のルールの妥当性に問題はないのではないかと考えている．

7. まとめ

スマートスピーカーに短文を話しかけた際の応答を記述する記述方式について検討し，提案した．従来だと単語に対する処理の定義しか行えなかったが，今回の提案内容により，短文の認識が可能になった．また，大学生を対象とした実験授業により，ルールの妥当性に関しては問題がないという結論に至った．

今後はこの記述方式をもとに，中学生や高校生でもこのルールを理解してオブジェクトの階層構造を想像しながらプログラムが組めるのかを検証したい．また，深いオブジェクト階層を持つ構造の学習や，オブジェクトの設計といった部分において今回の提案内容が優れているのかといった点を検証したい．

参考文献

[1] 本多佑希, 島袋舞子, 浅子秀樹, 兼宗進: スマートスピーカーのアプリケーション開発を支援するプログラミング学習環境の開発, 情報処理学会, 情報教育シンポジウム論文集, Vol.2019, pp.62-68, 2019.
[2] 大阪電気通信大学 兼宗研究室: プログラミング言語「ドリトル」, 入手先〈http://dolittle.eplang.jp/〉(参照 2020-10-15).
[3] NTT コミュニケーション科学研究所: 日本語語彙体系, 岩波書店 (1997).
[4] 情報処理学会: カリキュラム標準 J17, 入手先〈https://www.ipsj.or.jp/annai/committee/education/j07/curriculum_j17.html〉(参照 2020-10-15).
[5] 高井久美子, 渡辺博芳, 佐々木茂, 鎌田一雄: 個別学習と協調学習を組み合わせた授業例-オブジェクト指向モデリング導入教育における設計と実践-, 教育システム情報学会誌, Vol.28, No.3, pp.210-222, 2011.
[6] 兼宗進, 御手洗理英, 中谷多哉子, 福井眞吾, 久野靖: 学

```
果物=単語!作る.
果物:甘い?=「|A|クローバー！(A+"は甘いです") 話す」.
リンゴ=果物！作る.
レモン=果物!作る.
レモン:甘い？=「|A|
  クローバー!(A+"は甘くないです") 話す
」.
```

図 12 学生が課題1で作ったプログラム例1
Fig. 12 A student's program of practice 1.

```
OS ＝単語！作る.
OS：Linux？＝「｜ A ｜
  クローバー！(A ＋"は Linux") 話す
」.

Ubuntu ＝ OS ！作る.
Debian ＝ OS ！作る.
CentOS ＝ OS ！作る.

Windows ＝ OS ！作る.
Windows：Linux？＝「｜ A ｜
  クローバー！(A+"は Linux ではない") 話す
」.
mac ＝ OS ！作る.
mac：Linux？＝「｜ A ｜
  クローバー！(A+"は Linux ではない") 話す
」.
```

図 13 学生が課題1で作ったプログラム例2
Fig. 13 Another student's program of practice 1.

```
動物=単語!作る.
動物:場所？=「|A|
  クローバー！(A+"の場所は"+自分:住処+"です") 話す
」.
犬=動物!作る.
犬:住処="犬小屋".
猫=動物!作る.
猫:住処="こたつ".
熊=動物!作る.
熊:住処="洞窟".
```

図 14 学生が課題2で作ったプログラム例1
Fig. 14 A student's program of practice 2.

```
物=単語!作る.
物:材質は？=「|A|
  クローバー！(A+"の材質は"+自分:質+"です") 話す
」.
コップ=物!作る.
コップ:質="プラスチック".
靴=物!作る.
靴:質="革".
```

図 15 学生が課題2で作ったプログラム例2
Fig. 15 Another student's program of practice 2.

校教育用オブジェクト指向言語「ドリトル」の設計と実装, 情報処理学会, 情報処理学会論文誌プログラミング, Vol.42, No.SIG11, pp.78-90, 2001.