# BabaScript
## 人の行動をプログラムに組み込むためのプログラミング環境

馬場　匠見† 　橋本　翔† 　増井　俊之††

† 慶應義塾大学政策・メディア研究科　〒 252-0882　神奈川県藤沢市遠藤 5322
†† 慶應義塾大学環境情報学部　〒 252-0882　神奈川県藤沢市遠藤 5322
E-mail: †{bb,shokai}@sfc.keio.ac.jp, ††masui@pitecan.com

**あらまし**　コンピュータの動作の手順書としてプログラムが、人間の行動の手順書としてマニュアルやレシピといったものが存在する。プログラム上で人の行動を記述するためのプログラミング環境 BabaScript を提案する。BabaScript は、人への命令構文と値を返すことのできるクライアントアプリケーションを組み合わせることで、人オブジェクトをプログラム上で表現可能にするための一連の仕組みだ。

**キーワード**　ヒューマンコンピュテーション, プログラミング環境

## 1. はじめに

コンピュータの動作を制御するための手順書としてプログラムが存在する。世界中のコンピュータがインターネットを介して動作している実世界をプログラミングするために、センサ・アクチュエータを利用するようになっているプログラムが記述できる処理は増え続けており、プログラムが干渉可能な世界は大きく広がっている。

一方、人間の動作を制御するための手順書としては、レシピであったりマニュアルといったものが存在する。手順書に従うことによって、人間は適切・効率的に動作し、目的を達成できる。レシピやマニュアルの中身は、プログラムと同じようなものである。人に実行させたい処理が記述されており、人は記述内容を自分で解釈し、実行していく。例えば、料理のレシピには、

```
1 if 鍋が沸騰したら
2   パスタを鍋に投入する
```

というようなプログラムが、小売店の店員マニュアルでは

```
1 if レジに人が並んでいる
2   2番レジを開ける
```

といったプログラムが記述され、実行されている。

プログラムとレシピやマニュアルは、実行手順を表すという意味では同じものであるが、別の存在として認識されている。しかし、近年のコンピュータと人間の密接な関係を考慮すると、この状況は変化していくべきである。レシピやマニュアルの中では、コンピュータを扱う命令が書いてあることはよくあるプログラムの中で、計算資源として人間を使うことも提案されている人とコンピュータ、両方の世界を記述するための統一的なものは存在せず、両者を同じフォーマットのドキュメントとして記述することはできない。

本論文では、人の行動をプログラム内に記述可能にするプログラミング環境 BabaScript を提案する。人の行動をプログラム内で記述可能にすることで、同一フォーマットのデータ上で人とコンピュータ両方への命令を記述することができる。

## 2. BabaScript

機械と人をプログラム内で完全に同等に扱えるライブラリ BabaScript を開発した。BabaScript は、人への命令構文を含んだオブジェクト (以下、人オブジェクト) を宣言可能にするライブラリだ。通信用のライブラリを含んだクライアントアプリケーションと組み合わせることによって、人への命令とその実行結果の取得を実現する。

### 2.1 人への命令構文

人への命令構文を含んだオブジェクトを宣言し、そのオブジェクトに対してメソッド実行をすることで、メソッドと引数を元に人への命令内容を生成する。以下のようなプログラムで人への命令を記述する。

```
1 baba = new Baba.Script("baba");
2 baba.ボイルする (function(result){
3   ...
4 });
```

プログラム側で人オブジェクトを宣言するときには、第一引数に id を指定する。この id と同一の id を持つクライアントに対し、命令の配信が行われる。また、複数のクライアントアプリケーションが同一の id によって生成されている場合、script からの命令は特別な命令がない場合は、クライアントアプリケーションに分散して配信される。

人への命令メソッドの第一引数にオブジェクトを与えることによってクライアントアプリケーション側にオプションとして情報を送ることができる。特別なオプションとして broadcast が存在する。

```
1 baba.ボイルする ({broadcast: 5}, function(result){ ... });
```

上記のように、オプションに broadcast: num を指定することによって、同じ id を持つ全クライアントに対して同様の命

令を送信し、 num で指定した数だけ値が返ってきたらコールバック関数を実行することができる。

オプションの例としては返り値の型指定がある。プログラムへの返り値として様々な型が考えられるが、全ての型を考慮してプログラムを記述することは難しい。返り値の型を指定することによって、プログラム側で求めている値を人に入力させることが可能だ。例えば、数値の返り値を求める場合であれば、以下のようなプログラムを書き、クライアントアプリケーション側で数値だけを入力させるインタフェースを表示させることで実現する。

```
baba.数値を入力してください ({format: "int"}, function(
    result){ ... });
```

人への命令構文を含んだオブジェクトに対してメソッドを実行することによって、クライアントアプリケーションを組み合わせることで、人人は命令を処理し、その結果を入力すると、プログラムに自動で値が返る。

### 2.2 クライアント

ワーカー側では、命令を受け取り、値をプログラムに返すためのクライアントオブジェクトを生成する。このクライアントオブジェクトからのメッセージを受け取る関数を実装することによって、プログラマ側で自由にクライアントアプリケーションを実装できる。以下のようなプログラムでプログラムからのメッセージを受け取る。

```
client = new Baba.Client("baba");
client.on("get_task", function(order){
  プログラムからメッセージを受け取った時の挙動を記述する
});
```

メッセージを受け取る関数内において、ユーザに命令を伝え、処理結果を入力するようなインタフェースを生成してワーカーに提示する。また、クライアントオブジェクトが持つメソッド returnValue を使うことで、プログラムに処理結果を返すことができる。以下のようなプログラムが考えられる。

```
client.on("get_task", function(result){
  order = new Order(result.key)
  input = new Input(result.format)
  input.on("submit", function(value){
    client.returnValue(value)
  });
});
```

### 2.3 利用の流れ

本環境では、以下のような流れで利用される

（1） プログラムで人への命令構文を書く

（2） ネットワークを介して、適切な BabaScript Client に命令が配信される

（3） BabaScript Client は、命令をユーザに通知する

（4） 人が命令を処理する

（5） 人は処理結果を BabaScript Client に入力する

（6） 結果をネットワークを介して、プログラムに返す

（7） プログラムは、返ってきた値を元に指定された続きの処理を実行する

### 2.4 特　　徴

#### 2.4.1 人力をプログラムに溶けこませる

普通にプログラムを書いている中で、オブジェクトを扱っているのと同じような手法で人オブジェクトを利用可能である。人オブジェクトのメソッドを実行すれば、普通のオブジェクトと同じように、値が返ってくる。真の意味で、人と通常のオブジェクトは同じものとなる。

#### 2.4.2 あらゆる Web アプリに人力処理を組み込む

クライアントライブラリを読み込み、クライアントオブジェクトを生成するだけで、あらゆる Web アプリ上で人力処理のワーカーになることができる。拡張性がとても高く、既存のアプリケーションにも組み込むことができる

### 2.5 例

パスタ料理をつくるプログラムを以下に示す

#### 2.5.1 script 側プログラム

人への命令を記述するプログラムは、以下のようなものが考えられる。

```
baba = Baba.createScript("takumibaba")
baba.パスタ鍋に水を入れ沸騰させる ();
baba.on("boil", function(){
  baba.パスタ鍋にパスタを投入する ()
  baba.at("10␣minutes␣later", function(){
    baba.湯切りする ()
    baba.具材を痛めてたらフライパンにパスタを投入する ().適度に混ぜる (function(){
      baba.更に盛りつける ().done()
    });
  });
  baba.にんにくスライスを用意する ().鷹の爪を用意する ().炒める (function(){
    done()
  });
});
sensor = Sensor.create("鍋")
setTinterval(function(){
  if (sensor.getState === BOIL){ baba.emit("boil")}
}, 1000)
```

#### 2.5.2 client 側プログラム

人への命令を受け取り、ユーザに提示するプログラムは以下のようなものが考えられる。

```
<h1 id="title"></h1>
<button id="true-button"></button>
<button id="false-button"></button>
<script type="text/javascript">
  client = Baba.createClient("takumibaba").on("get_task",function(data){
    title = $("title")
    title.html(data.key)
    tButton = $("true-button")
    fButton = $("false-button")
    tButton.click(function(){
      client.returnValue(true)
    })
    fButton.click(function(){
      client.returnValue(false)
    })
  });
```

#### 2.5.3 クライアント

クライアントアプリは、以下のようなインタフェースを持つ。

## 3. 関連研究

計算機では処理できないようなタスクを解決するために、人を計算資源としてプログラムに組み込む手法はヒューマンコンピュテーション [?] と呼ばれ、様々な研究が行われている。

米 Amazon が運営している Amazon MechaicalTurk [2] は、クラウドソーシングのためのプラットフォームだ。

AUTOMAN [3] は、cloudprogramming という概念を唱え、通常のプログラミング言語内で、コンピュータによる計算と人による計算を統合した。CrowdForge [4] は、クラウドソーシングのための MapReduce 実装をした。クラウドソーシングするタスクを適切に分割し、人力で解かせた後、集合させるためのツールキットだ。jabbewocky [5] は、CrowdDB [6] では機械だけでは答えられないような DB へのクエリに対し、クラウドソーシングを使うことで返答させるための SQL ライクなプログラミングを提案している。CyLog [7] はクラウドソーシング/ヒューマンコンピュテーション系の研究は、人を計算資源として捉え、コンピュータの代替として人に処理を行わせることをやっている。本研究では、人の行動そのものを記述する。その中で、コンピュータの代替としての行動を記述することもある。

Human As Sensor なんてものもある。Using Stanger as Sensors では PRISM では

## 4. おわりに

## 文　献

[1] L. von Ahn. 2007. Human computation. In Proceedings of the 4th international conference on Knowledge capture. K-CAP '07. ACM.
[2] Amazon Machanical Turk http://www.mturk.com
[3]
[4]
[5]
[6]
[7]