BabaScript
人の行動を記述可能な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。BabaScript 環境によって、人・実世界・コンピュータの世界をより柔軟にプログラムすることが可能になる。

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
```

というようなことが、小売店の店員マニュアルには

```
1 if レジに人が並んでいる
2     2番レジを開ける
```

といった記述がされ、人によって実行されている。

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なプログラミング環境 BabaScript を提案する。BabaScript 環境によって、人の行動をプログラム内で記述可能となり、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。また、特殊なプログラミング言語を使わず、高い拡張性を持つため、様々なアプリケーションに組み込むことができる。

2. BabaScript

BabaScript 環境は、人への命令構文をプログラミング言語に付加するライブラリと、プログラムからの命令を受け取り、処理結果をプログラムに返すクライアントライブラリを組み合わせることで実現する。どちらのライブラリも、簡単に拡張・組み込みが可能となっている。既存のアプリケーションでも簡単に BabaScript 環境を導入することができる。

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 (以下、人オブジェクト) を宣言可能にするライブラリを javascript(node.js) と ruby の 2 言語で実装した。人オブジェクトを宣言し、そのオブジェクトのメソッドを実行をすることで、人への命令が可能となる。オブジェクトに定義されていないメソッドは全て人への命令として解釈され、メソッドと引数を元に人への命令内容が生成される。プログラム内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、特別なオプションがない場合は、命令は各クライアントアプリケーションに分散して配信される。以下のプログラムのような記述方法によって、人オブジェクトを宣言し、人に対して命令を送ることができる。

```
1 baba = new Baba.Script("baba");
2 baba.書類整理をする ({num: 5}, function(result){
3     ...
4 });
```

上記のプログラムの場合、メソッド名である”書類整理をする”と第一引数である ”num: 5” が命令としてクライアントアプリケーションに通知される。

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。

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

数値の他にも、文字列であったりリストの中から選択する、といったインタフェースが考えられる。

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

命令を受け取り、値をプログラムに返すための一連の機能をクライアント側のライブラリとして実装した。javascript,objective-c,java の 3 言語で実装されており、iOS や android などのモバイルデバイス上で動作する。クライアント側では、BabaScript 通信用のクライアントオブジェクトを宣言し、このオブジェクトを通すことでプログラム側との通信が可能となる。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ことができる。

```
1 client = new Baba.Client("baba");
2 client.on("get_task", function(order){
3     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
4 });
```

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必要がある。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。

```
1 client.on("get_task", function(result){
2     order = new Order(result.key)
3     input = new Input(result.format)
4     input.on("submit", function(value){
5         client.returnValue(value)
6     });
7 });
```

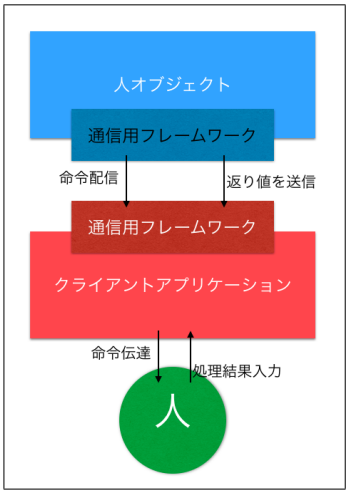

図 1　システム図

2.3 人力構文実行時の流れ

BabaScript 環境では、以下のような流れで人への命令が配信され、返り値を得ることができる。

（1） プログラムで人への命令構文を実行する

（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 アプリに人力処理を組み込む

クライアント用の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、どんなアプリ上でも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 サンプル

例えば、パスタ料理をつくるプログラムならば、以下のプログラム群のような記述方法が考えられる。

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")
list = ["ペペロンチーノ", "カルボナーラ", "トマトクリーム", "ボロネーゼ"]
baba.どれを作りますか ({format: "list", list: list},
    function(data){
  if(data.value === "ペペロンチーノ"){
    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 ()
      setTimeout(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 ()
        baba.適度に混ぜる (function(){
          baba.更に盛りつける ()
          done()
        });
      },1000*60*10)
      baba.にんにくスライスを用意する ()
      baba.鷹の爪を用意する ()
      baba.炒める ();
    });
  }
});
sensor = Sensor.create("鍋")
settInterval(function(){
  if (sensor.getState === BOIL){
    baba.emit("boil")
    clearInterval(arguments.callee)
  }
}, 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>
<div id="return-value-view">
</div>
<script type="text/javascript">
  client = Baba.createClient("takumibaba").on("get_task",function(data){
    title = $("title")
    title.html(data.key)
    $("#return-value-view").empty()
    if(data.format === "list"){
      select = $("select").addClass("value-list")
      for(int i=0;i<data.list.length;i++){
        option = $("<option>"+ list[i] +"</option>")
        select.append option
      }
      button = $("button").click(function(){
        client.returnValue($("value-list").val())
      });
      $("#return-value-view").append(select)
    }else{
      tButton = $("button").addClass("true-button").click(function(){
        client.returnValue(true)
      });
      fButton = $("button").addClass("false-button").click(function(){
        client.returnValue(false)
      });
      $("#return-value-view").append(tButton);
      $("#return-value-view").append(fButton);
    }
  });
</script>
```

2.5.3 クライアント

クライアントアプリは、命令が配信されていない時は待機用の画面を表示しておき、命令が配信された時に、その命令に応じて画面の表示内容を変化させる。

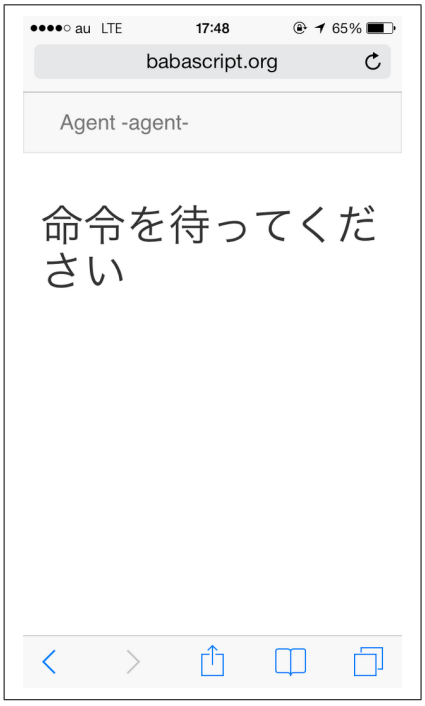

図 2　待機時の画面

図 3　リスト表示時の画面

3. 応　用　例

3.1 人の仕事や役割をプログラム化する

人の行動をプログラムとして記述可能になることで、人の仕事や役割がプログラム化・実行可能となる。仕事や役割はマニュアルやドキュメントという形で言語化されていることが多い。これらのマニュアル・ドキュメントをプログラム化し、実行できるようになれば、人はプログラムからの命令に従うだけで様々な仕事や役割を達成することができるようになる。ただ命令に従うだけなので、経験・引き継ぎは必要なく、人の代替が容易となる。また、全ての仕事はプログラムから管理されるため、人の運用効率の数値化などが可能となる。

3.2 柔軟な実世界プログラミング

人をセンサーやアクチュエータとして利用することで、現在一般的に使われているセンサーやアクチュエータでは実現困難な実世界プログラミングが可能となる。現在のセンサー技術では、その場の雰囲気を数値化・文字列化するなどのコンテキスト情報をうまく扱うことはとても難しい。また、アクチュエータも単一の動きに特化したものが多く、複雑な動きを実現することは難しい。BabaScript 環境でならば、プログラム上で人はセンサーやアクチュエータのオブジェクトと類似の挙動をすることができる。

人とセンサ・アクチュエータをうまく使い分けていくことが可能となる。コンテキスト情報を扱いそうなら人に、温度などの数値を取得するだけならばセンサーを使うとことができる。センサー・アクチュエータがその場に存在するならばセンサー・アクチュエータを動作させるが、ない場合は人に命令する、といったことも実現可能である。

4. 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 [1] 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 AmazonMechanicalTurk [2] は、クラウドソーシングのためのプラットフォームだ。mTurk API を通し、人間に対してタスクの実行を依頼することができる。AUTOMAN [3] は、crowdprogramming 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、MapReduce のような機能をクラウドソーシングのためのフレームワークだ。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、集合させるといったことができる。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 Datalog に似たヒューマンコンピュテーションのためのプログラミング言語だ。人をデータソースとしてプログラムの中で利用する手法を提案している。これらの研究は、人を計算資源・データソースとして捉え、コンピュータの代替として人を利用している。本研究では、人の行動そのものをプログラムとして記述し、実行可能なものにすることを目的としている。

ユビキタスコンピューティングの研究分野においては、Human as Sensor といった概念も存在しており、研究が行われている。MoboQ [8] では、場所ベースの Q&A サービスを実装し、その効果を検証した。Moboq ではプラットフォームとしてソーシャルメディアを利用しており、ソーシャルメディア上の人たちをセンサーとして利用している。スマートフォンを使ったセンシングのためのプラットフォームとしては、PRISM [9] などが発表されている。これらの研究では、人をセンサーとして利用し、情報を収集することを目的としている。本研究では、人の行動をプログラムとして記述することを目的としており、その利用方法はセンサーに限定されたものではない。

人のワークフローを定義する Web サービスとしては、atled [10] や Questetra [11] などが存在するが、これらのサービスは、人の行動をプログラムで記述するものではない。BabaScript 環境では、人・コンピュータの動作を同一のプログラム上で記述することが可能だ。

5. おわりに

5.1 今後の課題

今後は以下の問題について検討する。

5.1.1 命令に対する実行保証性

プログラムからの命令を人が必ず実行するとは限らない。命令が表示されるデバイスを見ていないことや、そもそも命令を無視するといったことも考えられる。実行保証性に関しては、今後検討していく必要がある。

5.2 結　　論

本論文では、人の行動を記述可能なプログラミング環境 BabaScript を提案した。人力処理構文と命令を受け取り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によって、プログラム上で人を表現することが可能になった。BabaScript 環境においては、人とコンピュータの双方は同じプログラム内で動きを定義することができる。今後は、課題の解決と有用性の検証を行う。

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] Barowy, D. W., Curtsinger, C., Berger, E. D., andMcGregor, A. AutoMan: A Platform for Integrating Human-Based and DigitalComputation. In Proc. OOPSLA (2012).

[4] A. Kittur, B. Smus, and R. E. Kraut. CrowdForge: Crowdsourc- ing Complex Work. Tech. Rep. CMU-HCII-11-100, Human- Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, February 2011.

[5] S. Ahmad, A. Battle, Z. Malkani, and S. Kamvar. The Jabberwocky Programming Environment for Structured Social Computing. In UIST, pp. 5364, 2011.

[6] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin. CrowdDB: Answering Queries with Crowdsourcing. In SIGMOD, pp. 6172, 2011.

[7] Atsuyuki Morishima, Norihide Shinagawa, Tomomi Mitsuishi, Hideto Aoki, Shun Fukusumi. " CyLog/Crowd4U: A Declarative Platform for Complex Data-centric"

[8] Liu, Yefeng and Alexandrova, Todorka and Nakajima, Tatsuo. Using Stranger As Sensors: Temporal and Geo-sensitive Question Answering via Social Media. Proceedings of the 22Nd International Conference on World Wide Web, pp. 803–814, 2013

[9] Das, Tathagata and Mohan, Prashanth and Padmanabhan, Venkata N. and Ramjee, Ramachandran and Sharma, Asankhaya. PRISM: Platform for Remote Sensing Using Smartphones. Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services. pp.63–76, 2010.

[10] https://www.atled.jp/

[11] http://www.questetra.com/