実世界プログラミングのための分散人力処理環境の開発
― 人をプログラミングする ―

1. 背景

コンピュータが得意なことはコンピュータが処理し、人は人にしかできないようなことや、人のほうが得意なことを処理するべきであるが、これは完全に実現されているわけではない。演算や単純な条件の判定などはコンピュータのほうが得意だが、コンテキストを伴った実世界情報の収集や、実世界への柔軟な干渉、人間とのコミュニケーションは人間のほうが得意である。しかし、人はコンピュータの方が得意な処理まで実行しているなど、お互いが処理すべき領域の分離ができていない。

これは、人が仕事や役割をこなそうとするときにも言えることである。仕事や役割における計算や条件判断にはコンピュータでも処理可能なことが多いが、コンピュータではなく人が処理した結果、ヒューマンエラーを引き起こしてしまうこともある。そういった仕事や役割の多くには、マニュアルやレシピが存在しており、人はマニュアルを参照し、記述内容を実行しようとする。マニュアルやレシピは、処理すべき内容が記述されているという点において、プログラムと同じ手順書である。人にとっての手順書であるマニュアルと、コンピュータにとっての手順書であるプログラム、双方ともに同じ手順書であるならば、同じフォーマットで記述可能であると考えられる。同じフォーマットでの記述が実現すれば、「コンピュータが得意なことはコンピュータがやり、人は人にしかできないようなことや、人のほうが得意なことをする」ことの実現が可能になり、お互いが処理すべき領域を明確にし、双方を対等な処理ノードとして扱えるような環境を構築可能となる。

2. 目的

本プロジェクトは、プログラム上において人とコンピュータの双方を対等に扱い、処理内容に応じて適切なノードに処理を行わせられるような仕組みの実現と、その仕組みを利用し、人の仕事や役割をプログラム化することを目的とする。

この目的を達成するために、人への行動指示をプログラム上で記述可能にする Babascript プログラミング環境を提案する。

加えて、上記環境を利用し、人の仕事や役割のプログラム化に取り組むことで 、Babascript の応用可能性について考察する。

3. 開発の内容

Babascript、Babascript Client、Babascript Manager、Babascript Plugins を開発した。

Babascript は、人への行動指示と指示に対する返り値を得ることのできるプログラミングライブラリだ。Node.js で動作するよう実装した。Babascript は、人への行動指示命令を取得し、タスクへと変換する人への命令構文と、分散処理基盤との通信を行う機能、指示に対する人の実行結果を返り値として受け取る機能、プラグイン機能などを有し、通常のプログラミング記法とほぼ同じ記法で人をプログラムに組み込むことができる。これによって、人を実世界とのインタフェースとして利用可能とな

る。例えば、図１のようなプログラムによって人をプログラム上に宣言し、行動指示命令を送ることができる。

```coffee
Baba = require "babascript"

baba = new Baba "baba"
baba.成果報告書を書く {}, (result) ->
  console.log result
  #result = {
    #value: "true"
    #task:
      #key: "成果報告書を書く"
      #name: "baba"
      #type: "eval"
      #cid: "1403501688\_0.2959382198750973"
      #at: 1403501771342
      #options: {}
    #getWorker: ->
      #return new Baba "baba"
  #}
```

```coffee
member = new Baba "2013mitoh"
member.成果報告書を書く {format: 'string', broadcast: 17}, (results) ->
  console.log results.length # 17
  for result in results
    # ...
```

```coffee
Baba = require "babascript"
UserManager = require "babascript-plugin-usermanager"

baba = new Baba "baba"
baba.set "manager", new UserManager()
# ...
```

図 1. Babascript プログラム例

Babascript Client は、Babascript からの命令を受け取り、命令を人に見える形に変換・提示し、返り値を入力させ、プログラムに返すための一連の機能を実装したライブラリとアプリケーションだ。Node.js, Web アプリ, Cordova アプリ, iOS アプリとして実装した。図２のようなプログラムで、Babascript からの命令を受け取ることができる。

```coffee
Client = require "babascript-client"
client = new CLient "baba"
client.on "get_task", (task) ->
  #... 命令を取得、処理を記述する
client.on "cancel_task", ->
  #... 命令がキャンセルされた時の処理を記述
```

図 2. Babascript Client プログラム例

インタフェースは図３のように実装した。命令の受信や返り値の送信をする機能とは分離した実装であるため、既存の UI だけでなく、他メディアをインタフェースとして利用する、といったことも可能だ。

図 3. Babascript Client インタフェース

Babascript と Babascript Client を組み合わせることで、プログラム上において人を表現し、その人に対する行動指示命令と返り値の取得が実現する。

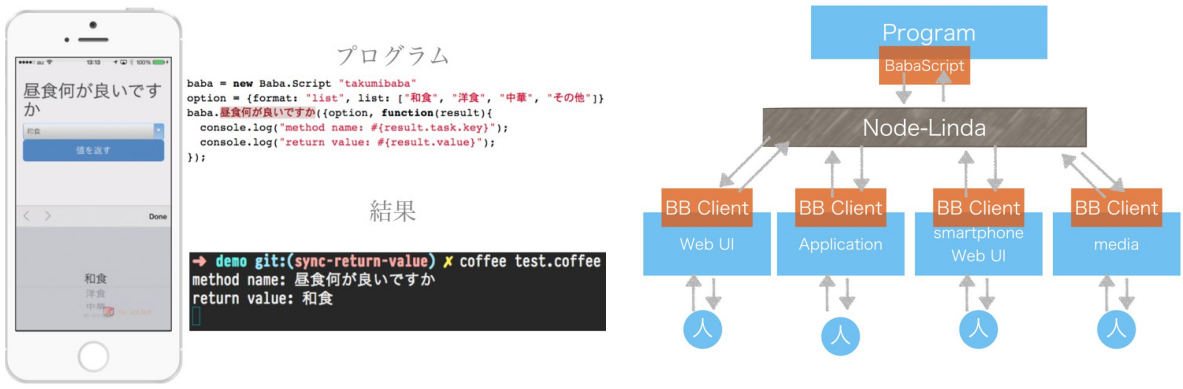

図 4. Babascript 全体像とシステム図

Babascript Manager は、Babascript で利用するための人力リソースを管理可能にする Web サービスだ。ユーザ・グループ情報の管理や、ユーザ情報のリアルタイム同期、タスクログの保存などの機能を有する。

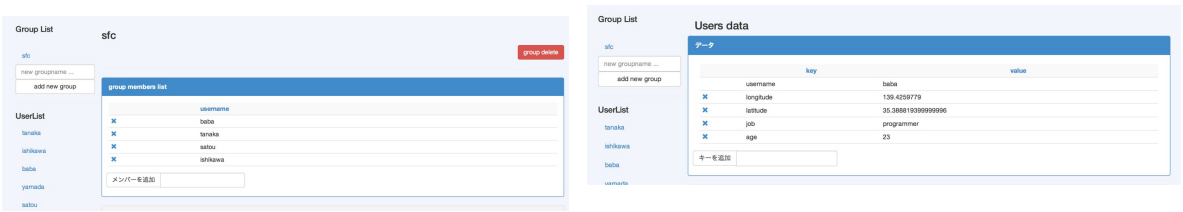

図 5. Babascript Manager Web インタフェース

Babascript は、プラグインによってその機能を拡張していくことができる。そこでプラグインの例として、UserManager プラグインと Logger プラグインの二つを実装した。UserManager プラグインは、前述の Babascript Manager と Babascript 及び Babascript Client を接続するためのプラグインだ。Logger プラグインは、人への行動指示命令や取得した返り値をログとして保存するためのプラグインである。プラグインは、指定されたメソッドを実装したオブジェクトを読み込むだけで簡単に Babascript 及び Babascript Client に組み込むことができる。

```
10  baba.data.users.get("baba").on "change_data", (data) ->
11    if data.level > 10
12      baba.data.users.get("baba").set "job", "master", {sync: true}
13      baba.研究してください {}, (result) ->
14        console.log baba.data.users.get(id).get 'job'
15        console.log result
```

図 6. Babascript プラグイン

また、Babascript 環境を利用して人の仕事や役割をプログラム化し、実行するといったことに取り組んだ。仕事や役割をプログラム化することによって、コンピュータにできることはコンピュータに実行させ、人は人にしかできないようなことを実行させることが可能となり、人への負担軽減やヒューマンエラー防止に繋げることができる。また、プログラム化対象を検討し、「論文を読む」「料理する」「ミーティング自動化」「スケジュール調整」といった、プログラムに近い仕事をプログラム化し、実行した。結果として、逐次的指示や作業の分割提示などが良いという評価を得るなどした。また、

人の処理の遅れがプログラム実行や処理継続に大きな影響を与えることや、命令の抽象度の適切な設計が必要であること、プログラムから命令されることに不安を覚える人がいる、といった知見を得る事ができた。

4. 従来の技術（または機能）との相違

Babascript 環境を利用することで、人をプログラム上で表現可能となり、通常のプログラムの記述方法とほぼ同じ記述方法で人力処理を実現することができる。また、人にフォーカスを当てていることで、人そのものの情報を利用したプログラムを記述することも可能である。

また、Babascript では複数人を対象として協調動作させるといったことも可能だ。普通のドキュメントやマニュアルでは、複数人の行動を協調的に操作するといったことは難しい。Babascript 環境では、様々な要素に応じて動的に行動指示を送ることができる。

5. 期待される効果

Babascript 環境を利用することによって、プログラムに簡単に人力処理を組み込むことができるようになる。今までは組み込みが難しかったような人間が関連した処理を簡単に実装することが可能だ。プログラム自体が干渉可能な領域が広がり、プログラムの可能性を大きく広げることとなる。

人の仕事や役割をプログラムとして記述可能になることによって、人はプログラムからの命令を聞くだけで様々な仕事や役割をこなせるようになると考えられる。作業進捗等もプログラムによって管理できることから、作業実行者の技能に応じて処理内容を変更したりすることも可能となる。特に、複数人を対象とした仕事を実行させる際には、各人の処理の進行具合、処理結果などをプログラム的に判断し、自動的に次の処理を実行させたり、処理を中断させたりといったことが可能となる。従来では人の判断が必要となったような場面においても、プログラムのみで処理を続行させることができ、人の作業負担の軽減や、ヒューマンエラー防止に繋がる。

6. 普及（または活用）の見通し

オープンソースプロジェクトとして公開し、全てのプログラマーが Babascript 環境をできるようにする。サンプルプログラムの拡充や勉強会・学会での発表を通して広く一般にアピールする。

7. クリエータ名（所属）

馬場　匠見（慶應義塾大学大学院 政策・メディア研究科）

（参考）関連 URL

http://github.com/babascript
http://babascript.org