端末を飛び出したオブジェクト: 分散プログラミングを活用した情報教育の提案*

兼宗 進 *, 中谷 多哉子 †, 御手洗 理英 §, 福井 眞吾 *, 久野 靖 *
筑波大学大学院 ビジネス科学研究科 *
(有) エス・ラグーン †　(株) アーマット §
〒 112-0012 文京区大塚 3-29-1*
{kanemune,fukui,kuno}@gssm.otsuka.tsukuba.ac.jp*
tina@slagoon.to†　rie@armat.com§

概要

教育課程の改定により、小学校から高等学校までの初中等教育において、プログラミングを含む情報教育の導入が進められている。筆者らは、初中等教育で活用可能な教育用オブジェクト指向言語「ドリトル」を開発し、現場での評価を行ってきた。本稿では、オブジェクトをネットワーク間で複製・共有して扱うドリトルの拡張について報告する。この機能により、ある生徒がドリトルのオブジェクトに名前を付けて公開したときに、他の生徒はそのオブジェクトを自分のプログラムに取り込んで再利用したり、共有して使うことが可能になった。中学校で行った実験授業と、企業の研修で利用した事例を報告する。

1　はじめに

近年ではインターネット環境の普及が急速に進んでいる。平成 15 年に公開された総務省の統計 [11] によると、平成 9 年末にわずか 6.4%だったインターネットの世帯普及率[1]は、平成 14 年末現在で 80%を超え、特に 13〜29 歳の若年層の利用率は 90%近くに達している。

このような背景から、今後の情報教育においては、スタンドアロンの計算機環境だけでなく、日常接する情報通信の中で計算機の果たす役割を理解することが重要になってくると考えられる。

筆者らはこれまで、「プログラミングの体験により計算機の動作を体験的に学ぶ」ための学習環境を提案し、教育用オブジェクト指向言語「ドリトル」の配布を行ってきた。[6][7][8]

本稿では、教育用のオブジェクト指向言語「ドリトル」の拡張により、オブジェクトをネットワーク環境で活用する学習環境を提案する。

2　ネットワーク教育に適したプログラミング言語

ドリトルの設計に当たっては、画面上の図形、タートル、GUI 部品などのオブジェクトにメッセージを送って操作する。[6] このモデルにより、初心者が興味を持ってプログラミングに取り組むことができ、オブジェクトを主体としたプログラミングを扱えることを示した。[8]

今回提案する分散共有ドリトルでは、ドリトルの入門用言語としての利点を維持しつつ、新たにネットワーク上でオブジェクトを扱う機能を加えることで、個人ごとのプログラミングだけでなく、他の生徒とコラボレーション作品を作成したり、ネットワークでの通信をプログラムを通して体験

An Proposal of New Information Education using Distributed Programming, Susumu Kanemune, Takako Nakatani†, Rie Mitarai§, Shingo Fukui*, Yasushi Kuno* (Tsukuba University*, S.Lagoon Co.,Ltd.†, Armat Corporation§)

[1] パーソナルコンピューターや携帯電話等の電子機器により、WWW の閲覧や電子メールを使用している個人のいる世帯の比率を示している。

することを可能にする。

ドリトルをネットワークに拡張するに当たり、次の特徴を取り入れることに留意した。

- ネットワークプログラミングの未経験者が扱えること
- プログラミングを通して、日常接する情報通信の原理を類推できること
- ネットワークプログラミングや分散プログラミングを学ぶことは目的としない。教員の適切な助言により、プログラミングを通して「なるほど、こういう原理で動いているのか」と思えることが重要と考える
- ドリトルのオブジェクトモデルを維持する。画面上のオブジェクトにメッセージを送り結果を視覚的に確認するモデルはネットワーク上のオブジェクトに対しても扱えるようにする

以下で、既存言語の分散共有機能を検討する。

LOGO[4] は教育用に設計された言語である。タートルグラフィックスにより、学習者は自分の行った操作を確認しながらプログラムを作ることができる。しかし、LOGO 自体はオブジェクト指向言語ではなく、タートルなどの画面上の操作対象をネットワークを介して扱う機能も提供されていない。

Smalltalk[1] の実装である Squeak[3] の上に構築された子供用のプログラミング環境として、SqueakToys[5] がある。SqueakToys では、画面上のパレットと呼ばれる領域に、値や制御構造を表すタイルを置くことでテキストの入力を不要としていることが特徴である。SqueakToys ではオブジェクトをプログラミングの基本とするが、ネットワークを介した操作を提供するものではない。

Java[2] はネットワークでの利用を考慮して設計されたオブジェクト指向言語である。Java には各種の分散オブジェクト指向環境が提供されている。しかし、教育を想定した言語ではないため、初心者が利用するには敷居が高いという問題がある。

スクリプト言語である Ruby[13] を分散環境に拡張したプログラミング環境として、dRuby[10] がある。サーバー上のオブジェクトにメッセージを送り実行するモデルは分散共有ドリトルと共通する部分が多いが、汎用言語である Ruby が画面に表示されないオブジェクトを基本とするのに対し、ドリトルでは画面上に表示されるオブジェクトを中心に操作する点が異なっている。

3 プログラミング言語「ドリトル」

ドリトル [6] [9] [12] は教育用に設計されたオブジェクト指向言語である。[2] 簡潔な日本語による構文を採用しており、オブジェクトに呼び掛ける形でプログラムを記述できる。

```
カメ太=タートル!作る。
カメ太!100 歩く 120 右回り 100 歩く 閉じる。
三角形=カメ太!図形にする（赤）塗る。
時計=タイマー!作る 1秒 間隔 10秒 時間。
三角形:ぐるぐる=「時計!「!36 右回り」実行」。
実行ボタン=ボタン!作る。
実行ボタン:動作=「三角形!ぐるぐる」。
```

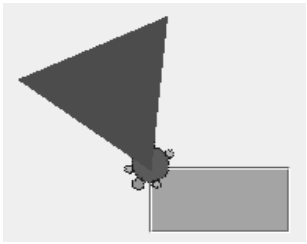

図 1: ドリトルのプログラムと実行例

図 1 に、ドリトルのプログラム例を示す。以下ではこの例を使い、ドリトルの構文を解説する。

カメ太=タートル!作る。

“カメ太” というタートルオブジェクトを作る。画面にはカメの姿をしたタートルオブジェクトが現れる。タートルオブジェクトはタートルグラフィックスを実現するオブジェクトであり、軌跡の線を描きながら画面を移動することができる。

カメ太!100 歩く 120 右回り 100 歩く 閉じる。

ドリトルでは、“!” でオブジェクトに呼び掛ける形でプログラムを記述する。命令を続けて書くことで、前の命令の結果であるオブジェクトに、次の命令を続けて送ることができる。実行すると画面に正三角形が描かれる。

三角形=カメ太!図形にする (赤) 塗る。

カメ太に “図形にする” を送り図形オブジェクトを作る。[3] そして、図形オブジェクトに “塗る”

[2] 最新版はプログラミング言語「ドリトル」のサイト (http://www.logob.com/dolittle/) から入手できる。

[3] タートルが描いた線は、“図形にする” で明示的に切り

を送って色を塗り、“三角形” という名前を付けている。

時計=タイマー!作る 1 秒 間隔 10 秒 時間。

タイマーは決められた時間の間、一定間隔でプログラムを実行するオブジェクトである。ここでは “時計” という名前のタイマーオブジェクトを作り、実行時間と実行間隔をセットしている。

三角形:ぐるぐる=「時計!「!36 右回り」実行」。

オブジェクトには、メソッド (埋め込まれた手続き) を定義できる。ここでは、三角形に “ぐるぐる” という名前のメソッドを定義している。このメソッドが実行されると、時計は “「!36 右回り」” を 1 秒間隔で 10 秒間 (すなわち 10 回) 繰り返し実行する。

実行ボタン=ボタン!作る。

“実行ボタン” という名前のボタンオブジェクトを作っている。画面には、ボタンの形をした GUI 部品が現れる。

実行ボタン:動作=「三角形!ぐるぐる」。

“実行ボタン” に、マウスで押されたときに実行するメソッドを定義している。[4] 画面に表示された “実行ボタン” を押すことにより、三角形に定義された “ぐるぐる” というメソッドが実行される。その結果、三角形が画面上で 1 秒ごとに 36 度ずつ回転するアニメーションが表示される。

4 ネットワーク機能

4.1 基本的な考え方

ドリトルでは、画面上のオブジェクトにメッセージを送り、操作の結果を視覚的に確認するモデルにより、初心者がオブジェクトを単位としたプログラミングを行える。また、グラフィックスや GUI 部品を使用したプログラミングを体験することを通して、日常接するさまざまな計算機の中でプログラムが動作していることを学ぶことができる。

ドリトルのオブジェクトは、変数 (プロパティ) や配列に格納し、取り出して使うことができる。分散共有ドリトルでは、オブジェクトの格納をネットワーク上に拡張する形で、初心者がオブジェクトの転送を容易に扱えるようにした。分散共有ドリトルのプログラミングを体験することを通して、日常接するさまざまなデータ通信の中で、プログラムが動作していることを学ぶことができる。[5]

分散共有ドリトルの実行時には、ネットワーク上にオブジェクトサーバーのプロセスを起動しておく。オブジェクトサーバーには、名前を付けてドリトルのオブジェクトを登録することができる。登録したオブジェクトは、複製と共有という 2 通りの活用を行えるようにした。

オブジェクトの複製 (図 2) は、サーバからオブジェクトの複製を取り出す。取り出されたオブジェクトは通常のローカルなオブジェクトとなる。この機能はサーバを介して複数のドリトル環境 (クライアント) 間でオブジェクトや値をやりとりするために用いる。

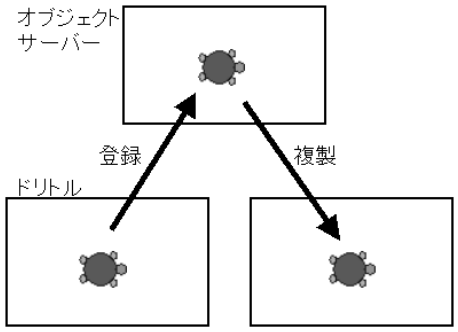

図 2: オブジェクトの登録と複製

オブジェクトの共有 (図 3) は、サーバ上のオブジェクトを共有する形で取り出す。この場合、取り出されるのはサーバ上のオブジェクトを参照する「共有オブジェクト」となり、共有オブジェクトに対する操作は、そのサーバ上のオブジェクトに対応するすべての共有オブジェクトに影響を及ぼす。

以下では、登録、複製、共有の機能を順に説明する。

放さない限り、タートルオブジェクトの一部になる。カメの尻尾が長く伸びた姿を想像すると理解しやすい。

[4] いくつかのオブジェクトには、特別な名前のメソッドを定義することにより、ある条件が満たされたときにそのメソッドを実行することができる。たとえばタートルや図形オブジェクトでは、“衝突” という名前のメソッドを定義しておくことで、他のオブジェクトとぶつかったときの動作を記述できる。

[5] 分散共有ドリトルでは、「ネットワークプログラミング技術の習得」を主要な目的とはしていない。そこで、ソケット通信などの習得が難しいと思われるモデルは採用しなかった。

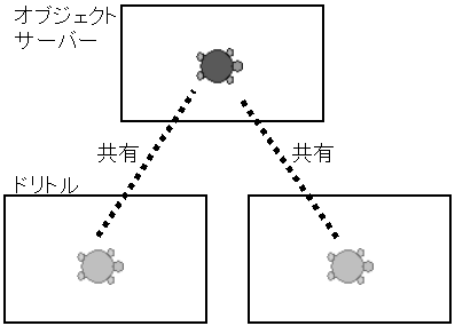

図 3: オブジェクトの共有

4.2 オブジェクトの登録と複製

図 4 に、オブジェクトサーバーにオブジェクトを登録・複製するプログラム例を示す。

```
// ローカルオブジェクトの生成 (1)
カメ太=タートル!作る。

// サーバーへの接続 (2)
サーバー!"sv1" 接続。

// オブジェクトの登録 (3)
サーバー!"kame1"（カメ太）登録。

// オブジェクトの複製 (4)
カメ吉=サーバー!"kame1" 複製。

// オブジェクトの操作 (5)
時計=タイマー!作る
時計!「カメ太!10 歩く。カメ吉!15 歩く」実行。
```

図 4: オブジェクトの登録と複製

(1) で、タートルオブジェクトを生成する。(2) で、サーバー “sv1” に接続する。引数には IP アドレスまたはホスト名を指定できる。(3) で、サーバーにオブジェクト “カメ太” を “kame1” という名前で登録する。(4) で、サーバーから “kame1” を複製し、カメ吉という名前にする。(5) で、画面上の 2 つのオブジェクト (カメ太とカメ吉) を同時に動かす。サーバーから複製したオブジェクトは、ローカルのオブジェクトと区別することなくメッセージを送り操作することが可能である。

4.3 オブジェクトの共有

図 5, 図 6 にプログラム例を示す。

クライアント A で図 5 を、クライアント B で

```
// サーバーへの接続 (1)
サーバー!"sv1" 接続。

// オブジェクトの共有 (2)
カメ助=サーバー!"kame1" 共有。
```

図 5: オブジェクトの共有 (1)

```
// サーバーへの接続 (1)
サーバー!"sv1" 接続。

// オブジェクトの共有 (2)
カメ助=サーバー!"kame1" 共有。

// オブジェクトの操作 (3)
時計=タイマー!作る
時計!「カメ助!10 歩く」実行。
```

図 6: オブジェクトの共有 (2)

図 6 を実行することを考える。(1) で、サーバー “sv1” に接続する。(2) で、サーバー上のオブジェクト “kame1” を共有し、“カメ助” というローカルの名前にする。(3) で、クライアント B から共有オブジェクトを操作する。クライアント B の共有オブジェクトに送られたメッセージはサーバー上のオブジェクトに転送され、実行が行われる。サーバー上のオブジェクトの状態が変化すると、クライアント A, クライアント B の共有オブジェクトも画面上で位置の変化などを反映する。

5 実装

5.1 分散共有ドリトルの構成と動作環境

ドリトルの処理系は Java 2 [2] により記述されており、Java 2 が動くさまざまな環境で動作する。表 1 に、ドリトルの提供するオブジェクトを示す。

図 7 にドリトルの実行画面を示す。実行画面と編集画面を画面上部のタブで切り替えて操作する。画面下部には実行などのボタンが並んでいる。

分散共有ドリトルの機能を使うときは、ネットワーク上でオブジェクトサーバーを起動しておく。オブジェクトサーバーは画面を持ったプロセスである。ドリトルと同一のホスト上で動作してもよく、また、ネットワーク中に複数存在してもよい。

図 8 にオブジェクトサーバーの実行画面を示す。

表 1: ドリトルのオブジェクト

オブジェクト	説明
タートル	タートルグラフィックスの機能を提供するオブジェクト
図形	図形を表すオブジェクト。タートルに“図形にする”メソッドを送ることで生成される
GUI 部品	ボタン、ラベル、フィールド、リスト、選択メニューを表すオブジェクト
配列	一連の値の並びを表すオブジェクト
タイマー	ブロックを指定した“実行”メソッドにより、一定間隔で一定時間、操作を実行するオブジェクト
数値	数値を表すオブジェクト
文字列	文字の並びを表すオブジェクト
論理値	真偽値を表すオブジェクト
色	色を表すオブジェクト
ブロック	動作の列を表すオブジェクト
サーバー	オブジェクトサーバーとの通信を行うオブジェクト
シリアルポート	外部機器を制御するオブジェクト

図 7: ドリトルの実行画面

モニター画面には、サーバーに登録されたオブジェクトが表示される。

オブジェクトサーサーバーはドリトルと似た画面インターフェースを使用しているが、プログラムを編集・実行するインターフェースは用意されていない。また、画面上のオブジェクトは表示されるだけであり、画面上のボタンを押してもメッセージは送られない。

サーバーに登録されたオブジェクトは、ドリトルから確認することができる。

6 実施した授業

6.1 中学校での実験授業

中学校で実験授業を行った。授業は放課後を利用し、2 年生の生徒 3 人が参加した。コースは 1

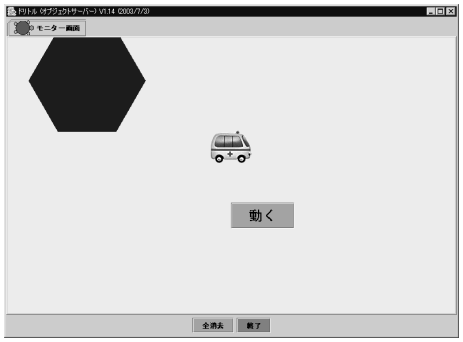

図 8: オブジェクトサーバーの実行画面

時間を目安とし、一週間の間隔で 2 回行った。テキストは筆者が用意し、教員が講義を担当した。

生徒は全員がワードプロセッサや電子メールを使用した経験があり、基本的なキーボードやマウスの操作は問題なく行えた。プログラミングについては、1 学期に 5 時間行われたドリトルの授業の中で、全員がタートルグラフィックス、メソッド、ボタンの概念を学んでいた。それ以前にプログラミングの経験のある生徒はいなかった。表 2 に実施したカリキュラムを示す。

表 2: 実験授業のカリキュラム

授業	ねらい	内容
1	オブジェクトを転送するプログラミング	ネットワークの説明 接続 オブジェクトの登録 オブジェクトの複製
2	オブジェクトを共有するプログラミング	数当てクイズプログラム 石取りゲームプログラム

説明の後、生徒には自作のオブジェクトを作ってもらい、各人のオブジェクトをサーバーに登録した。図 9 に生徒が作成したプログラムを示す。

全員が自作のオブジェクトをサーバーに登録した後、他の 2 人のオブジェクトを自分のドリトルに取り込んで実行した。図 10 に生徒のプログラム例を示す。このプログラムでは、サーバーに「カメゾウ」という名前のタートルオブジェクトを登録した後、他の生徒が登録したタートルオブジェクトとボタンオブジェクトを取り込んでいる。取り込んだオブジェクトがオブジェクトの性質を持つことを確認するために、タートルオブジェクトの「三角」メソッドを実行したときに画面に三角

```
// 生徒 A
サーバー！”192.168.1.24” 接続。
カメゾウ＝タートル！作る 。
カメゾウ:星＝「「!100 歩く 144 右回り」!5 回 繰り返す」。
サーバー！”カメゾウ星” (カメゾウ) 登録。

// 生徒 B
サーバー！”192.168.1.24” 接続。
カメ太＝タートル！作る ”rocket.gif” 変身する。
カメ太:三角＝「「!100 歩く 120 右回り」!3 回 繰り返す」。
サーバー！”カメ太三角” (カメ太) 登録。

// 生徒 C
サーバー！”192.168.1.24” 接続。
変身＝ボタン！”変身” 作る。
変身：動作＝「カメゾウ！”99.gif” 変身する」。
サーバー！”変身ボタン” (変身) 登録。
```

図 9: 生徒の登録プログラム

形が描かれることと、ボタンオブジェクトを押下したときにタートルの画像が変化することを確認した。

```
サーバー！”192.168.1.24” 接続。
カメゾウ＝タートル！作る。
カメゾウ:星＝「「!100 歩く 144 右回り」!5 回 繰り返す」。
サーバー！”カメゾウ星” (カメゾウ) 登録。

カメ太＝サーバー！”カメ太三角” 複製。
カメ太！三角。
変身ボタン＝サーバー！”変身ボタン” 複製。
```

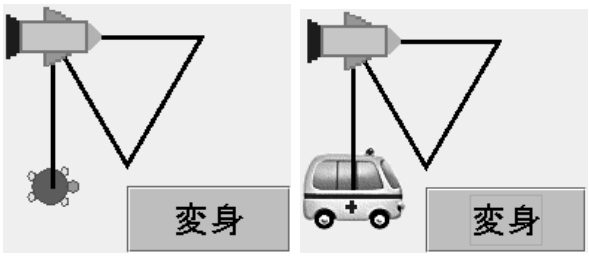

図 10: 生徒の複製プログラムと実行例 (右図はボタン押下後)

次に、石取りゲームを扱った。これは、「石の山から交互に数個ずつ石を取り合い、最後の 1 個を取ったほうが負け」というゲームである。今回は、最初の石の数を 10 個とし、同時に取れる石は 1〜3 個というルールとした。全員が図 11 のプログラムを実行し、石を取って戻す操作を順番に行うことにより、ゲームを行った。

続いて、図 11 のプログラムについて、ゲーム

```
// 画面に部品を並べる
もらう＝ボタン！”もらう” 作る 0 100 位置。
とる＝ボタン！”とる” 作る 0 50 位置。
表示＝フィールド！作る 0 0 位置。
おくる＝ボタン！”おくる” 作る 0 -50 位置。

// サーバーにつなぐ
サーバー名=”192.168.1.24”。
サーバー！(サーバー名) 接続。

// サーバーに「山」を登録する。石の数を表す
山=10。
サーバー！”山” (山) 登録。

// ゲームを作る
もらう:動作=「山=サーバー!”山” 複製。表示！(山) 書く」。
とる:動作=「山=(山 − 1)。表示！(山) 書く」。
おくる:動作=「サーバー!”山” (山) 登録。表示！”” 書く」。
```

図 11: 石取りゲーム

の動作と照らし合わせながら生徒と読み合わせを行った。

このゲームではネットワーク上のひとつの資源 (石の数を表す変数) に複数のプログラムがアクセスを行うが、生徒から「石を取って戻したのに、数が正しく減らないことがあった。他の人と同時に操作したからだろうか」という質問があり、議論になった。例として、「サーバー上に 10 個の石が存在するとき、2 人の生徒が同時にその石を取り、それぞれ 2 個と 3 個減らしてサーバーに戻すと、サーバー上には何個の石が残るか」という問いかけをしたところ、「5 個」という予想が出た。そこで生徒にプログラムの動作を口頭で発言してもらいながら、黒板で動作のシミュレーションを行い、8 個になる場合と 7 個になる場合があることを検証した。

このように、ネットワーク上でデータを共有する体験により、1 台の計算機だけのプログラミングでは気づかない問題を発見し、議論の中で深めていくことが可能であることがわかった。

6.2 企業での新入社員教育

企業の研修でドリトルを使用し、プログラミングの授業を行った。コース名は「コンピュータの動作原理」である。対象となった生徒は、営業希望または SE 希望の新入社員 15 人である。プログラミング経験は、研究等で使用していた生徒は 2

名、大学の授業で体験した生徒が 2 名、経験のない生徒が 11 名という内訳であった。

生徒は前日までの 2 日間でパーソナルコンピューターをキットから組み立てる講習を受講し、今回の 3 日間の授業に臨んだ。計算機の原理のうち、ハードウェアについては前日までに部品の構成を学んでいることから、ドリトルを使い計算機がソフトウェアで動いているという側面を体験的に学ぶことを授業の中心とした。

表 3 にカリキュラムを、図 12 に授業風景を示す。1 日目の午前は、計算機の構成を説明した後、ドリトルの動作に必要なソフトウェアとして、Java とドリトルのインストールを行った。1 日目の午後は、プログラミングの体験として、タートルグラフィックスを扱った。オブジェクトに命令を送ると実行が行われるという基本的な考え方を、画面上のオブジェクトの移動を確認しながら学習した。2 日目の午前は、タイマーによるアニメーションを題材に、繰り返しとスレッドの概念を学んだ。2 日目の午後は、ボタンが押されたときの動作とタートルが他のオブジェクトと衝突したことの検知を題材に、オブジェクトのメソッド定義を扱った。続いて、教室内の LAN 環境と各端末がネットワークで接続されていることを説明し、以下の分散共有ドリトルの機能を扱った。

- オブジェクトの転送。押すと何らかの動作を行うボタンを作り、サーバーに登録する。続いて、他の生徒のボタンをサーバーから複製し、端末上で実行する
- オブジェクトの共有。サーバー上のタートルオブジェクトを全端末から共有し、生徒が作ったプログラムから一斉に操作する

3 日目の午前は、各自の作品プログラムを作成する時間とした。3 日目の午後は、発表の準備として PowerPoint 等でスライドを作った後、発表会を行った。

表 4, 表 5 に授業の前後に行ったアンケートの結果を示す。以下は、結果から読み取れる内容である。

- (1) 全員が情報通信を支えるソフトウェアの存在を理解した (強い肯定が 67%から 100%に増加)
- (2) 楽しさを感じ (肯定が 40%から 93%に増加)、(4) 難しいという印象が減少した (肯定が 80%から 46.7%に減少)
- (5) プログラムは論理で書かれており、正しく書けば動く (強い肯定が 67%から 93%に増加) が、(3) きちんと書かないと動かない (強い肯定が 67%から 80%に増加)

(1) は情報通信の中で使われるソフトウェアの存在を問う設問である。授業前は弱い肯定が見受けられたが、授業後は全員が強い肯定に変化したことから、分散共有ドリトルによるプログラミングを体験することで、「WWW や電子メールといった情報通信がソフトウェア技術に支えられている」ことを体験的に学べることを確認した。

表 3: 新人研修のカリキュラム

時間	内容
1 日目 (AM)	計算機とソフトウェアの関係。Java とドリトルのインストール
1 日目 (PM)	はじめてのプログラミング
2 日目 (AM)	タイマーによる繰り返し
2 日目 (PM)	メソッド定義、ネットワーク
3 日目 (AM)	作品製作
3 日目 (PM)	作品発表会

図 12: 新人研修の授業風景

7 まとめ

本論文では、教育用オブジェクト指向言語「ドリトル」を拡張し、ネットワーク上でオブジェクトを活用することが可能な分散共有ドリトルを提案した。今後は、分散共有機能を活用したコラボレーションによる作品作りや、排他制御等を含む情報教育での利用について研究を進めていきたい。

表 4: 理解度アンケートの結果 (授業前)

設問		肯定 ←			→ 否定	平均
		4	3	2	1	
(1)	WWW や携帯電話の裏ではプログラムが動いている	66.7%	26.7%	6.7%	0.0%	3.60
(2)	プログラミングは楽しい (楽しそう)	6.7%	33.3%	33.3%	26.7%	2.20
(3)	プログラムはきちんと書かないと動かない	66.7%	26.7%	6.7%	0.0%	3.60
(4)	プログラミングは難しい (難しそう)	40.0%	40.0%	20.0%	0.0%	3.20
(5)	プログラムは正しく書けばそのとおりに動く	66.7%	26.7%	6.7%	0.0%	3.60

表 5: 理解度アンケートの結果 (授業後)

設問		肯定 ←			→ 否定	平均
		4	3	2	1	
(1)	WWW や携帯電話の裏ではプログラムが動いている	100.0%	0.0%	0.0%	0.0%	4.00
(2)	プログラミングは楽しい (楽しそう)	26.7%	66.7%	6.7%	0.0%	3.20
(3)	プログラムはきちんと書かないと動かない	80.0%	13.3%	6.7%	0.0%	3.73
(4)	プログラミングは難しい (難しそう)	13.3%	33.3%	46.7%	6.7%	2.53
(5)	プログラムは正しく書けばそのとおりに動く	93.3%	6.7%	0.0%	0.0%	3.93

授業に協力頂いた、藤枝市立西益津中学校と (株)リコーの皆様に感謝致します。

参考文献

[1] Adele Goldberg and David Robson. *Smalltalk-80: The Language and Its Implementation*. Addison-Wesley, 1983.

[2] James Gosling, Bill Joy, and Guy Steele. *The Java Language Specification*. Addison-Wesley, 1996.

[3] Dan Ingalls, Ted Kaehler, John Maloney, Scott Wallace, and Alan Kay. Back to the future: the story of Squeak, a practical Smalltalk written in itself. In *Proceedings of the 1997 ACM SIGPLAN conference on Object-oriented programming systems, languages and applications*, pp. 318–326, 1997.

[4] Seymour Papert. *Mindstorms : children, computers, and powerful ideas*. Basic Books, 1980.

[5] John Steinmetz. Computers and squeak as environments for learning. In *Squeak: Open Personal Computing and Multimedia*. Prentice Hall, 2001.

[6] 兼宗進, 御手洗理英, 中谷多哉子, 福井眞吾, 久野靖. 学校教育用オブジェクト指向言語「ドリトル」の設計と実装. 情報処理学会論文誌, Vol. 42, No. SIG11(PRO12), pp. 78–90, 2001.

[7] 兼宗進, 中谷多哉子, 井戸坂幸男, 御手洗理英, 福井眞吾, 久野靖. 教育用オブジェクト指向言語「ドリトル」による授業実施とその評価. 情報処理学会情報教育シンポジウム (SSS2002), pp. 229–236, 2002.

[8] 兼宗進, 中谷多哉子, 御手洗理英, 福井眞吾, 久野靖. 初中等教育におけるオブジェクト指向プログラミングの実践と評価. 情報処理学会論文誌, 印刷中.

[9] 紅林秀治, 兼宗進, 岡田雅美, 佐藤和浩, 久野靖. 画面を飛び出したオブジェクト: 自立型ロボットを活用した情報教育の提案. 情報処理学会 情報教育シンポジウム (SSS2002), 2002.

[10] 関将俊. dRuby による分散オブジェクトプログラミング. アスキー, 2001.

[11] 総務省. 平成 14 年通信利用動向調査の結果. http://www.soumu.go.jp/s-news/2003/030307_1.html.

[12] 中谷多哉子, 兼宗進, 御手洗理英, 福井眞吾, 久野靖. オブジェクトストーム: オブジェクト指向言語による初中等プログラミング教育の提案. 情報処理学会論文誌, Vol. 43, No. 6, pp. 1610–1624, 2002.

[13] まつもとゆきひろ, 石塚圭樹. オブジェクト指向スクリプト言語 Ruby. アスキー, 1999.