人文科学とコンピュータ　49－8
（2001．1．19）

自発的機能を持つオブジェクトの設計と実装

駱 福全　小林 郁典　渡邉 勝正

奈良先端科学技術大学院大学 情報科学研究科

本稿では、我々は自発的機能を持つオブジェクトの設計と実装について報告する。自発的機能を持つオブジェクトの枠組は”人間の自発性”を応用した設計である。我々はオブジェクトに自発的な振舞いを持たせるために、関連規則と自発的なオブジェクトモデルを提案する。システムの開発者は依頼の分析段階で、ある依頼に対して行われる自発的な振舞いを、関連規則で予め定義しておく。さらに、オブジェクトの設計段階で、設計した自発的なオブジェクトモデルのルールクラスのメソッドに、”if 文”の形式で定義された自発的な振舞いを記述する。この方法論を検証するために、この方法論に従った”Web 探険ソフトウェアシステム”を実装した。このシステムは、あるホームページのリンク先ホームページを自発的に取得する機能と、取得したWebホームページを自発的に分類する機能を持っている。

キーワード：人間の自発性、自発的機能、関連規則、ルールクラス、依頼、Web 探険ソフトウェアシステム

Design and Implementation of Objects with a Spontaneous Function

LO Fuchuan, KOBAYASHI Ikunori and WATANABE Katsumasa

Graduate School of Information Science
Nara Institute of Science and Technology

In this paper, we discuss designing and implementing a framework of objects with a spontaneous function. The framework of objects with a spontaneous function is a design in which "Spontaneity of the human" is applied. To give the object spontaneous behavior, we propose rules of relevance and a spontaneous object model. The developer of the system defines spontaneous behavior beforehand by rules of relevance for a certain request at the analysis stage of the system. In addition, the spontaneous behavior is described in the form of "If then" as the method of a class of rules of a spontaneous object model which designed at the design stage of the system. To verify this methodology, "Web exploration software system(WSS)" according to this methodology is implemented. WSS has provided the spontaneous functions to acquire the home pages from a certain home page ahead and to classify the acquired Web home pages.

Keywords: spontaneity of the human, spontaneous function, rules of relevance, class of rules , request, WSS

1 はじめに

年々複雑かつ大規模化しているソフトウェアシステム開発の生産性を高めるためには、対象世界および問題領域をいかに把握するか、すなわち問題領域をどのようにモデル化するかが重要である。このモデル化手法の一つとして、オブジェクト指向方法論がより自然なモデル化を行なう技術として注目されつつある。ソフトウェアシステムにオブジェクト指向を用いるメリットは、その開発や利用が人間の思考形態に近い形で行えることだと言われている[2][3]。

しかしオブジェクト指向の範囲内で知的処理や自発的機能を必要とするソフトウェアシステムを開発する場合には、かなり特殊な技術が必要になる。これはオブジェクト指向の基本要素であるオブジェクトは、メッセージを受けとることで自らの持つ操作を起動する受動的なモジュールであることと、知的処理で必要なルールをオブジェクトの中へ自然に融合させることが難しいことになると考えられる[4]。

本研究では、このような難しいことを解消する一つの試みとして、ソフトウェアシステム開発者が知的なシステムをより自然に構築するには、"人間の自発性"を応用すれば、よいのではないかと考えた[5]。この発想からわれわれはオブジェクト指向の範囲内でオブジェクトが自発的な振舞いをするための関連規則の定義方法とルールクラスの設計方法を提案する。システムの開発者は分析段階で依頼に対して、関連規則として、行うべき自発的な仕事を決定する。さらに、設計段階で提案した自発的機能のモデルで事前条件を判定して、実行するメソッドを定める。実行するメソッドは、依頼の意味を広く解釈して、依頼に関連したことも含めて、気を利かせて行う。

最近の関連した研究として例えば、並行オブジェクトの能動的性質について述べたものがある[6]。そこでは、オブジェクトが能動的に条件判断をするモデルになっているが、本論文では、依頼を受けたときに自発的に行動を起こすモデルになっている。

本論文では、自発的機能のモデルにおいて着目する性質について述べ、この性質を実現する方法を考察し、自発的な振舞いを持つ知的システムを実装する。さらに前述の設計方法に基づき、知的システムの開発において、対象の本質に開発者の意識を集中させることで、分析から実装までのよりスムーズな移行を目指した設計方法について述べる。

2 自発的機能の定義

まず、次のような言葉の定義がある。(広辞苑)[1]

- 自発性 (spontaneity):思考活動および行為において、他から教示され、または、影響されるのではなく、内部の原因・力によって思考、行為がなされること。
- 自発的 (spontaneous):自分から進んでするさま。

次に、我々は自発的機能を厳密に定義するために、次のような二つの側面から、アプローチする。

2.1 人間の側面から

ある種の生物は、同じ刺激に対して常に同じように反応する。しかし、我々人間を始めとする多くの生物は、何らかの経験を持つ場合と持たない場合とでは、同様の状況の下で異なったことが行うことが多い。その理由は、「個人学習能力の差」にあると考えられる。学習とは子供の頃から、親、学校や社会などの環境から、いろいろな刺激や知識を受けて、疲労などのように一時的なものを除いて、経験に転換して「こころ」に蓄積してきたものである。この学習してきた結果はものを判断したり、行動を行ったりする一連の知的活動が行うためのベースとなる。ところで、年齢、教育程度などの「個人学習能力の差」によって、異なった知的活動を行なわれることは当然である。この一連の知的活動が行われるわけは人間がある目標を達成するために、人間の内部（心）から知的活動がなされることによる。これを「人間の自発性」と定義する。

2.2 システムの側面から

近年、システムの構成は複雑かつ知的な方向へ進んでいる傾向が見られている。ここで、この「人間の自発性」をソフトウェアシステムに応用するときに、ソフトウェアシステムにおける「自発的機能」はどう表現すればよいかが問題になる。この問題に対して、2.1 節で述べたように「個人学習能力の差による異なった自発性」の個人差を取り除かなければならないようになる。これを取り除いて自発的機能を行うプロセスを明確に表すために、次のような四つのモジュールを考案した。この四つのモジュールは学習した結果 (ルールク

ラス) に基づいて構成され、実行される。

1. 事前条件判断モジュール：仕事の依頼 (request) を受け入れるがどうかを事前に判断する。
2. 計画モジュール：受け入れた依頼を実行する計画を立てる。
3. 計画選択モジュール：複数の計画から一つの計画を選択する。
4. 計画実行モジュール：選択した計画を実行して、結果を返す。

以上の四つのモジュールで、自発的機能を行うために不可欠なプロセスの静的関係を表している。四つのモジュールのモデルの詳細について、後の3章で述べる。

3 自発的機能を持つオブジェクトの概要

自発的機能を持つオブジェクトは、従来のオブジェクト手法の特徴をほぼ踏襲したままで自発的な振舞いを実現するオブジェクトである。システムの開発者にとって、このオブジェクトを利用した設計の枠組は依頼の分析段階と自発的なオブジェクトの設計段階とに分けられる。

3.1 依頼の分析段階

システムの開発者は何らかの依頼に対して、依頼の意味を広く解釈して、依頼に関連したことも含めて、気を利かせて行うべきことを予め分析して、関連規則として定義しておく。

3.2 関連規則

依頼 Req について、関連する仕事 func(あるいは process) を規則 (rules) で定義する。

定義 1：関連規則の定義形式

Req ::= *func_1* + *func_2*　選択 (choice)
　　| *func_1* & *func_2*　逐次 (serial)
　　| *func_1* | *func_2*　並列 (parallel)

Req が与えられると、関連規則から、行なうべき仕事を推論して選定する (infer_func)。

例 1：インタネットのホームページ (HP) から URL を抽出 = テキストタイプの判定 & (HTML の場合 | Java Script の場合 | XML の場合).

例 2：電子メールの受信 = テキストの復号 & ウイルスのチェック & 受信者のメールリストへの登録.

例 3：関数 f(x) の微分 = 関数 f が x で連続微分可能の判定 & (微分する + 不連続点の種類を知らせる).

関連の定義を階層的に行なって, 抽象化や具体化をすることもできる.

例 4：配列 A の平均 = 配列 A の次元を判定 & (1 次元配列 A の平均 + 2 次元配列 A の平均).

2 次元配列 A の平均 = A 全体の平均 | 行毎の平均 | 列毎の平均.

それぞれの仕事 (func) には, 前条件 (pre-condition) が付けられていて, Req に対して, 選定されるかどうかの判断を与える。

```
if(pre_condition){ body_part } ;
```

仕事の依頼と選定：関連規則による選定

(1) 仕事を依頼する側：

```
Req(arg_list); /* 依頼 Req を出す */
```

(2) 仕事を選定する側（自発性の発揮）：

```
func_set = infer_func(Req);
/* Req の関連規則から、 前条件を判定して、
実行する関数を決める */
exec(func_set,arg_list); /* 与えられた引数で
選定した関数を実行する */
```

関連規則は、次のように展開されて、関数を選定して、実行 (exec) される。

```
選択：if(pre_condition1) select(func_1);
        else
        if(pre_condition2) select(func_2);
逐次：if(pre_condition1) select(func_1);
        if(pre_condition2) select(func_2);
並列：para
        if(pre_condition1) select(func_1);
        if(pre_condition2) select(func_2);
        endpara
```

例 5：Req が「HP から URL を抽出」(例 1) の場合、依頼側と実行側は次のようになる。

依頼側：ホームページを取り出して、URL の抽出を依頼する。

```
url = get_next_URL();
new_file = load_HP(url);
get_new_URL(new_file);
```

実行側：与えられ HP のファイルから新しい URL を抽出する。

```
file_type = check_text_type(new_file);
para
  if (file_type == HTML)
    find_URLinH();/* for HTML */
  if(file_type == Java_Script)
```

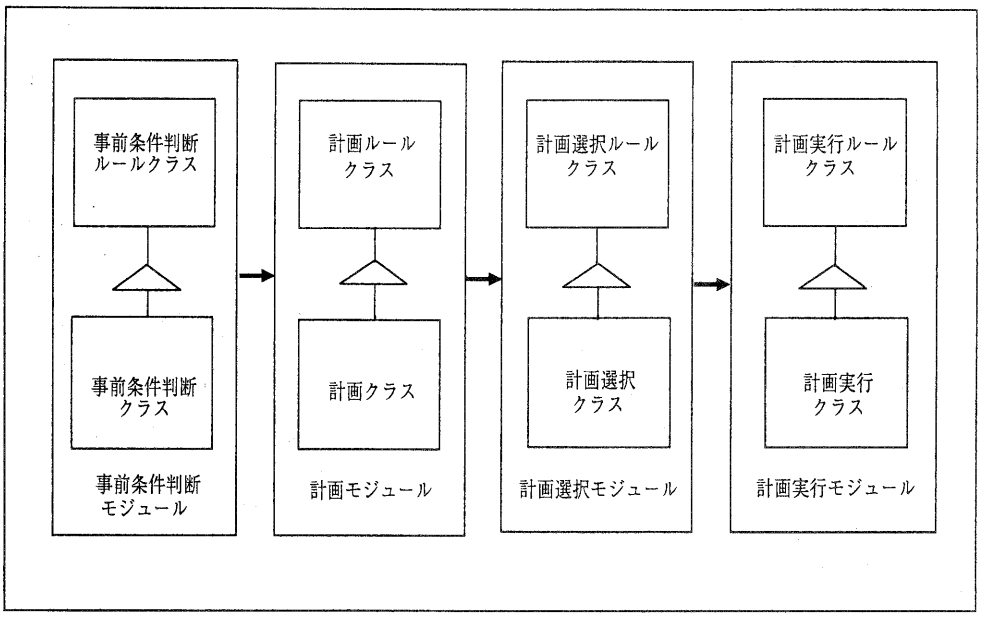

図 1. 自発的機能のオブジェクトモデル

```
    find_URLinJ();/*for Java_Script*/
  if(file_type == XML)
    find_URLinX();/* for XML */
endpara
```

3.3 自発的なオブジェクトの設計段階

ある依頼に対して行われる自発的な振舞いを前述した関連規則で解釈できたら、オブジェクトモデルでの表現は図 1 に示しているようなモデルで考えられる。このモデルは以下の四つのモジュールからなる。

- 事前条件判断モジュール
  定義：ある依頼を受け入れるかどうかの判断を行うモジュールである。
  構成：事前条件判断クラス は 事前条件判断ルールクラスを継承して、一つのモジュールになる。
- 計画モジュール
  定義：関連規則 (3.2 節) を見て、いろいろな計画を立てるモジュールである。
  構成：計画クラス は 計画ルールクラスを継承して、一つのモジュールになる。
- 計画選択モジュール
  定義：複数の計画の中から、一つの計画を選択するモジュールである。
  構成：選択クラス は 計画選択ルールクラスを継承して、一つのモジュールになる。
- 実行モジュール
  定義：選択した計画を実行するモジュールである。
  構成：計画実行クラス は 計画実行ルールクラスを継承して、一つのモジュールになる。

ある依頼に対して、四つのモジュールは時間とともにステップ 1 からステップ 4 まで次のように動く。

1. 事前判断の結果、依頼を受け入れたら、次の計画モジュールに送る。
2. 複数の計画を立てて、それを次の計画選択モジュールに送る。
3. 一つの計画を選択して、それを実行するように、次の計画実行モジュールに送る。
4. 指示された計画を実行する。

ステップ 1 からステップ 4 までの動きによって、一つの自発的機能が行われることになる。

四つのモジュールにルールクラスを付加することで、自発的に振舞うことができる。また、一般的なオブジェクトと同じように、メッセージを受け取ることで、メソッドを実行に移すこともできる。それぞれのモジュールの基底クラスはそれぞ

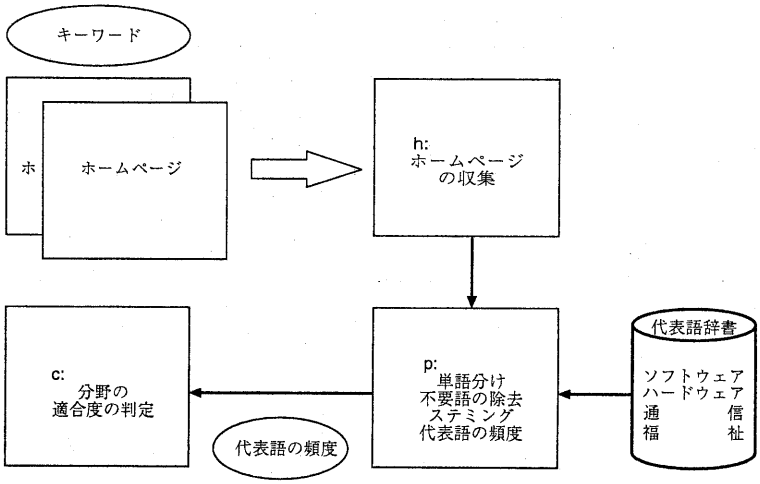

図 2. ホームページの収集と分類

れのルールクラス を継承することによって、ルールクラスのメソッドを使えることになる。それぞれのルールクラスの役割は次の節で述べる。

3.4 ルールクラスの役割

ルールクラスの実行メソッドは、プロダクション・ルール（条件-行動の対）に基づいて記述される。それぞれのルールは、システムがユーザーからの依頼に対して、自発的な振舞いのサービスを行うためのルールである。ルールの拡張と変更は他のオブジェクトに影響しないし、しかも容易に行えるため、ユーザーに対する自発的なサービスの柔軟性を一層高めることができる。さらに、システムの開発者は、知的なシステムを設計する段階で、知的な処理の問題に必要なルールを簡潔に考えられる。

4 自発的機能の実装例

4.1 WSS(Web 探険ソフトウェアシステム)

4.1.1 システムの機能

- 分野を表すキーワードによって、 Web ホームページを収集する。
- その分野に適しているかどうかを判定する。

4.1.2 システムの構成

- h:開始のホームページから，リンクを辿ってホームページを収集するプロセスである。
- p:テキストの単語分け・不要語の除去・単語のステミング・代表語の頻度の計算を行なうプロセスである。
- c:分野の適合性を判定するプロセスである。

以上のプロセスから成る（図 2)。

4.1.3 システムの自発的機能の分析

表現形式の異ったホームページの収集や、分野の判定の依頼を自発的機能としての関連規則として表すと、次の依頼 1 と依頼 2 のような分析結果になる。

- HP から URL を抽出（依頼 1） = テキストタイプの判定 & (HTML の場合 | Java Script の場合 | XML の場合).
- HP の分野を判定（依頼 2） = ホームページの言語を判定 & (代表語の重み法 | 内積法).

4.1.4 システムの自発的機能の設計

”依頼 1”の分析結果に基づいて、3.3 節で述べた自発的機能のオブジェクトモデルを用いて、自発的機能の設計を行うと、次のような設計（図 3）になる。

1. 事前条件判断モジュールでは、ファイルのタイプがテキストタイプかどうかの判定をする。この判定はファイル（ホームページ）単位で行う。
   以下の 2~4 はテキストの 1 行単位で行う。
2. 計画モジュールでは、表現形式の異なったテキストタイプのファイル（HTML の場合、JavaScript の場合と XML の場合）から新しい URL を抽出するための複数計画を提供する。
3. 計画選択モジュールでは、表現形式の判定結果（複数）から一つを選択する。
4. 計画実行モジュールでは、表現形式に従って

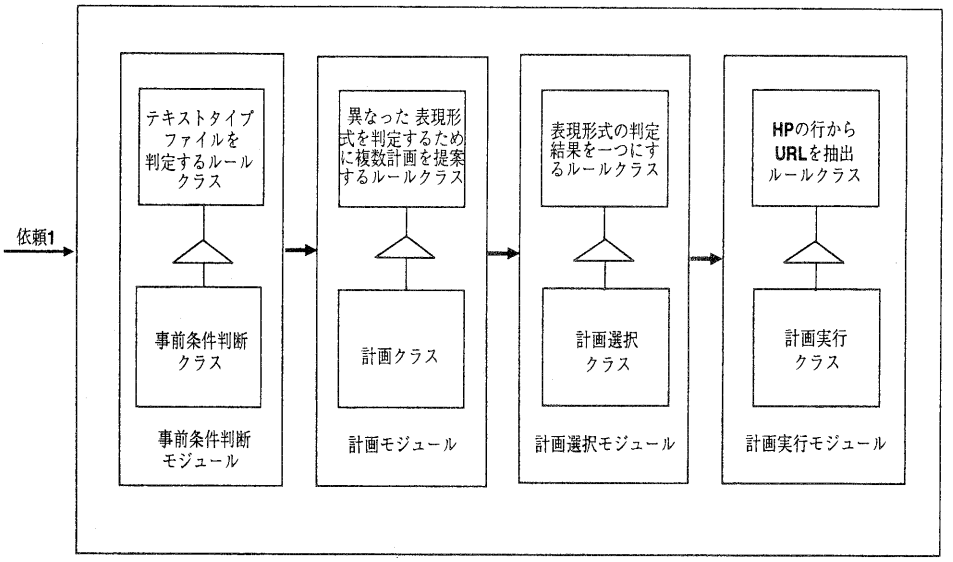

図 3.WSS の自発的機能のオブジェクトモデル

新しい URL を抽出する。

同じように"依頼2"の自発的機能の設計は、次のようになる。オブジェクトの構成図は図3と同形である。

1. 事前条件モジュールでは、ホームページ言語の判定をする。(たとえば、英語がどうか)
2. 計画モジュールでは、代表語の重み法と内積法により分野の判定を提案する。
3. 計画選択モジュールでは、それぞれの分類方法で判定した結果を見て、適切な分野かどうかを判断する分類結果を選択する。
4. 計画実行モジュールでは、選択された分類結果を外部インタフェースに出す。

4.1.5 ルールクラスのルール表現

ここで、依頼1の"HTML の場合のプロダクション・ルール"の表現は次のように考えられる。

```
public String find_URLinH(String line)
{ // URL のフラグの検出
    apos=line.indexOf("<a href=");
    if(apos < 0 )
      {apos=line.indexOf("<A HREF=");}
    if(apos < 0 )
      {apos=line.indexOf("<A href=");}
    if(apos >= 0) // http:の確認
     {apos=line.indexOf("http:",apos);}
    if(apos >=0 ) // URL の区切りの検出
      { bpos=line.indexOf('"',apos);
        if(bpos >= 0)
          { // URL の取り出し
            if(apos < bpos)
                .
        URL=line.substring(apos,bpos);
        // URL="http://www.ibm.com/"
            }
        }
    return URL;
}
```

4.2 図書サービス

図書サービスは、図書館や企業の社内図書室などで行われている図書の新着処理や返却処理というサービスである。ここでは、"図書の新着処理"という依頼に対して、関連規則で自発的機能の分析を行う。

4.2.1 図書の新着処理の自発的機能の分析

図書の新着処理 = 図書の登録重複のチェック &(図書の登録 | 代金支払い手続き | 予約者へ通知).

4.2.2 図書の新着処理の自発的機能の設計

上の分析結果に基づいた、自発的機能のオブジェクト設計は次のようになる(図 4)。

1. 事前モジュールでは、同じ図書があるかどうかを判定する。なければ、その判定結果を次のモジュールに移す。

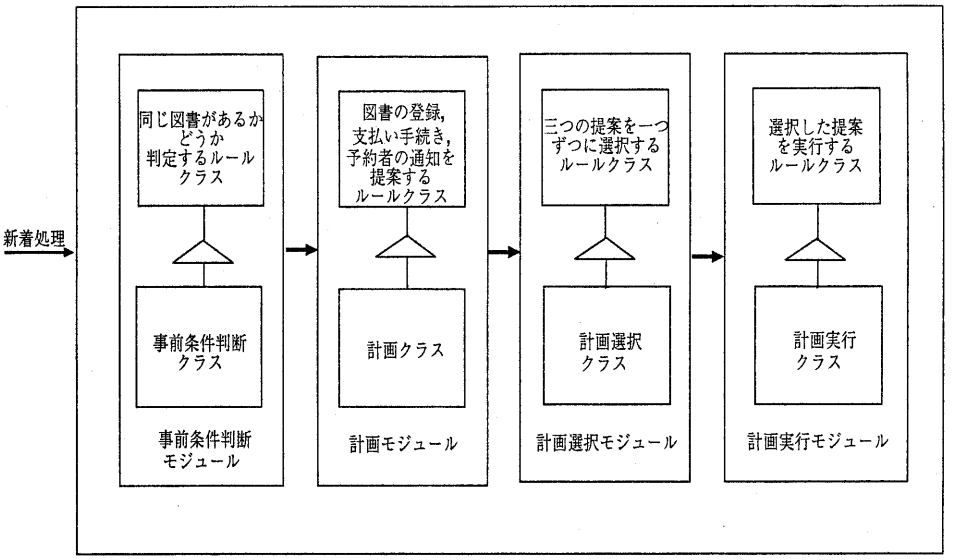

図 4. 図書の新着処理の自発的機能のオブジェクトモデル

2. 計画モジュールでは、図書の登録、支払い手続きと予約者のへ通知を提案する。
3. 計画選択モジュールでは、図書の登録、支払い手続き、予約者への通知の三つの提案を一つずつ選択する。
4. 計画実行モジュールでは、選択した三つの提案を逐次あるいは並行に実行する。

5 おわりに

本稿では、自発的機能を持つオブジェクトの設計と実装について述べた。自発的機能を設計するために、"人間の自発性"を参考にした。そこで、人間の自発性はソフトウェア工学のオブジェクト指向の領域において、どのようなモデルで表現できるかと考えた。それによりオブジェクト分析に適用できる関連規則とオブジェクト指向設計に適用できる自発的機能のオブジェクトモデルを提案した。

提案の有効性を示す実装の例として、WSS と図書サービスを取り上げた。従来の手続き型プログラムより、プログラムの変更・拡張などを行いやすい。プログラムの柔軟な拡張性が高まったと思われる。

今後の問題として、(1) 関連規則の動的更新に対応できるような枠組の提案 (2) 自発的機能に基づいた設計法の検証 (3) 様々な問題への適応、などがある。

謝辞

日頃からご討論いただく、本学の堀山貴史助手、中西正樹助手および渡邊研究室の皆様に感謝します。本研究は、一部文部省科学研究費補助金 基盤研究 (B)(2)11480068 によるものです。

参考文献

[1] 新村出：広辞苑 (第四版), 岩波書店 (1993).

[2] 本位田真一, 山城明宏：オブジェクト指向システム開発, 日経 BP 出版センター (1997).

[3] ジェームズマーチン, ジェームズ J. オデル：オブジェクト指向方法序説,(三菱 CC 研究会 OO タスクフォース訳), トッパン (1995).

[4] Joseph P. Bigus, Jennifer Bigus : Constructing Intelligent Agents with Java, WILEY(1998).

[5] 駱福全, 小林郁典：エージェントに基づいた利用者支援システムの開発環境, 平成 10 年度電気関係学会四国支部連合大会講演論文集,pp.309(1998).

[6] 上田賀一, 平井譲：並行システムのオブジェクト指向設計に適したプログラミング言語の開発, 情報処理学会論文誌,Vol.41,No.7,pp.1965-1975(2000).