知識工学と人工知能　59－17
（１９８８．７．７）

ソフトウェア把握支援のための説明機能

山田宏之†　　手塚慶一††
† 愛媛大学工学部　†† 大阪大学工学部

本稿では，ソフトウェアのソースコード内の手続きに対して，予め記述された手続きのドキュメント情報（機能情報と構造情報）から手続きの内容に関する説明を生成する機能について考察する．但し，説明生成の対象をオブジェクト指向型言語で記述された待ち行列シミュレーションソフトウェアとし，さらに，ある機能を実現している手続き（メソッド）はより簡単な機能をもつ手続きの組合せで実現されていると仮定する．本説明機能はメソッドをその構造に基づいてトップダウン的に自然言語により説明する．機能情報の表現形式として格フレームを採用し，メソッドの構造に対応して定義された洗練化規則を用いて，説明文を洗練化することにより，柔軟な表現にすることを試みている．本機能は手続きに対する概略的な説明をユーザに示すことにより，ユーザのソフトウェアに対する把握を支援できると考えられる．

An Explanation Facility for Supporting Software Grasp

Hiroyuki YAMADA† and Yoshikazu TEZUKA††

† Faculty of Engineering, Ehime University, 3 Bunkyo-cho, Matsuyama-shi, 790 japan
†† Faculty of Engineering, Osaka University, 2-1 Yamadaoka, Suita-shi, 565 japan

This paper presents an explanation facility for functions of procedures. This facility uses document, which we call functional information and structural information, of the software. We supposes that object-level software (queueing simulation software) is described in Object Oriented Language and that the procedure (method) is constructed by more primitive procedures. This facility explains the software in topdown fashion down to primitive functions or known functions. We use the case frame as representation of the functional information. The explanation shown by the system is easy to read because it is refined by means of rules, which are defined each procedure's structure (sequence, conditional branch and iteration).

1．はじめに

ソフトウェア開発において，既存のソフトウェアが簡単に再利用できると最初から開発する手間が省略できるために開発効率向上が期待できる．しかしながら，他人が開発したソフトウェアを把握することは容易でない．また，自分の要求に合うようにソフトウェアを変更する場合にしばしば一部の修正が別の箇所に影響を及ぼすことがあり，その対処に多大な労力を強いられている．

我々はソフトウェアの再利用（継承）の効率を向上させるために，ソフトウェアの変更に伴う他の部分への影響を変更に伴う相互作用と呼び，相互作用を管理するための情報を知識ベース化する枠組みについて検討し，変更時にユーザに相互作用の波及箇所を指摘できるシステムの構成について検討してきた［１］．システムがユーザに変更箇所を示唆しても，システムが指摘する相互作用の波及箇所は絶対でないので，ユーザがその箇所の機能を知らないために修正の必要性に気が付かない可能性がある．

このような事態を防ぐにはユーザの対象ソフトウェア把握を支援する必要がある．ソフトウェア把握支援の一手段として，システムがソフトウェアの機能についての概略を説明することが考えられる．すなわち，概略的な説明は，細かい部分を利用者から隠すので，ソフトウェア把握が支援できる．

ソフトウェアの機能を説明をするとき，システムが利用可能な情報としてソフトウェアのソースコードがある．しかしながら，ソフトウェアは処理手順（ＨＯＷ情報）が記述されており，ソフトウェアの意図（ＷＨＡＴ情報）が陽に記述されていないため，ソースコードを直接説明のために用いることは得策ではない．

そこで，本稿ではソフトウェアのソースコードで用いられる手続きに対して，予めその手続きの機能に関する付加情報（ドキュメント情報）を記述し，この情報を利用してソフトウェアの内容に関する説明を生成する機能について検討する［２］．但し，説明生成の対象をオブジェクト指向型言語で記述された待ち行列シミュレーションソフトウェア［３］とし，さらに，ある機能を手続き（メソッド）がより簡単な機能をもつメソッドの組合せで実現されていると仮定する．

2．ドキュメント情報

システムがそのシステムのもつ対象領域の知識，ならびに，システムの振舞いについて説明するときに，対象領域の構造や機能に関する知識を利用することにより，詳細な説明を生成することができる［４］．ソフトウェアの説明を生成するための情報として，本稿ではソフトウェアを構成する手続きの働きに関する情報（機能情報）とソフトウェアの内部における手続きの論理構造（構造情報）を利用する．これらの情報は，ドキュメント情報の一部とみなすことができる．

ソフトウェアの機能情報は，手続き（メソッド）が何をするかについて記述されたもので，そのメソッドに対する実現の方法については記述されない．

構造情報は，手続きがもつ論理構造である．本稿では，メソッドが持つ構造は，逐次，条件分岐，反復構造（これらを基本構造と呼ぶ）ならびにこれらの入れ子構造とする．

3．ソフトウェアの説明生成の概要

3．1　説明に対する要求

ソフトウェアの機能に関する説明に対する要求として以下のものが挙げられる．

（ａ）プログラマの対象ソフトウェアの既知度に応じた説明．

（ｂ）読解性のよい説明．

（ａ）は，システムがユーザの能力を把握し，それに対応した説明を生成することであり，（ｂ）は，ユーザの直感に訴えるために，ユーザのソフトウェアの対象領域で用いられる用語で明瞭な説明文を生成することである．

（ａ）の要求項目の問題を解決するには，システム内にユーザの能力，理解度を表現するモデルを獲得する必要があり，高度な知識獲得機能が要求され，現在その実現法について検討している．本稿では，ユーザが対象ソフトウェア記述言語を理解しているレベルとし，ユーザモデルの構築は考えていない．また，説明の詳細さのレベルとして，①単に手続きの機能のみの説明を生成するレベル，および，②手続きを構成する下位手続きの機能情報とその手続きの構造とから説明を生成するレベルの２つを用意している．前者は，変更支援システムにおいて変更箇所と示唆された手続きの概略的な機能の説明であり，後者はその手続きがどの様に構成されているかについての概略的な説明である．後者の説明の中に現われる下位手続きの中で，未知の手続きがあればそれに対して繰り返し後者の説明生成を適用することにより，ソフトウェアをトップダ

ウン的にプリミティブレベルの手続きまであるいは既知の手続きまで分解するしながら，ソフトウェアを説明する．

（ｂ）の要求項目の問題を解決するためには，柔軟な自然言語生成機能が必要である．そこで，自然な説明文を生成するために，機能情報を意味表現形式で表現し，説明時に意味表現から自然言語文に変換する．この時，手続きの構造情報を文脈情報と捉え，この情報に基づいて説明文を洗練化する．次節以降では，機能情報の意味表現形式として格文法を採用し，格文法に基づいた日本語生成ならびに手続きの構造に応じた洗練化規則に基づく説明文の洗練化について検討する．

３．２　機能情報からの説明文生成

まず，オブジェクト指向型言語で記述された待ち行列シミュレーションソフトウェアにおいて，モデルを記述するために利用されるメソッドに対して機能情報を記述した．このとき，ユーザの理解を支援するために，対象領域であるシミュレーションに現れる用語を使用するように留意した．例えば，メソッド "defineArrivalSchedule"に対する機能情報は，“シミュレーションオブジェクトが確率密度分布関数に基づいて規定の時間間隔でシミュレーションモデルに到着する．”である．ただし，現段階ではソフトウェア開発者が機能情報を最初に記述する．

次に，記述した機能情報に現れる格情報を分析した結果，主格，対象格，方法格，場所格，時格が最小限必要であることがわかった．さらに，場所格として始点格と終点格が，時格として開始格，終了格，時刻格，時間格がある．図１に，格に対応した助詞，及び助詞相当語句を示す．さらに，機能情報に利用されている動詞を図２に示すように概念的な意味で分類した．ここで，メソッド 'defineArrivalSchedule'の機能情報を，格フレーム形式で表現した例を図３に示す．この機能情報において動詞は'到着する'は in_out_verbに属し，主格の格情報は 'シミュレーションオブジェクト'，方法格の格情報は '確率密度関数'，時間格の格情報は '規定の時間間隔'，終点格の格情報は'シミュレーションモデル'である．

機能情報を格文法に基づいて表現した場合，格フレーム表現を説明文に変換する必要がある．そこで，機能情報から説明を生成するために動詞決定規則と規則選択パターンとから構成される説明生成規則を用いる．

格文法により表現された情報からの，日本語文の生成の手順について述べる．

```
主　格　－　は
対象格　－　を
方法格　－　で
終点格　－　へ
始点格　－　から
開始格　－　から
終了格　－　まで
時点格　－　に
時間格　－　間
```

図１　格の種類

(1) シミュレーションの制御に関する動詞
(control_verb)
　　開始する，中断する，再開する，終了する
(2) シミュレーションへの出入に関する動詞
(in_out_verb))
　　到着する，出る
(3) 物事の行為に関する動詞
(action_verb)
　　定義する，初期設定する，計画する，
　　続ける，準備する，実行する
(4) 資源の取捨に関する動詞
(resource_verb)
　　解放する，獲得する
(5) 資源，インスタンスの生成に関する動詞
(creation_verb)
　　作る，生成する
(6) さまざまな問い合わせに関する動詞
(query_verb)
　　質問する，尋ねる

図２　動詞の分類結果

```
(in_out_verb (到着する)
    (主格　(シミュレーションオブジェクト))
    (方法格　(確率密度関数))
    (時間格　(規定の時間間隔))
    (終点格　(シミュレーションモデル)))
```

図３　格フレームの一例

1） 格文法により表現された機能情報と規則選択パターンとの照合をとり，規則選択パターン内の変数に格情報を代入する．

2） 1）の結果に基づき動詞決定規則を選択し，概念的に分類した動詞とその動詞に対応する助詞を決定する．

ここで，規則選択パターンは，パターン内の変数に対応する格を必須格と自由格に分類し，自由格の種類により動詞決定規則を選択するために用いられる．また，動詞決定規則は，規則選択パターンに代入された自由格の種類により，動詞とその動詞に対応した助詞を決定するための規則である．シミュレーションへの出入に関する動詞を例に説明する．規則選択パターンには，必須格として，主格，方法格，時間格が，自由格として場所格がある．規則選択パターンに基づいた動詞決定規則の一例を図４に示す．

概念的な意味で分類した場合，同じ格情報をもつとき，動詞を一意に決定できなくなるという問題が生じる．一例として，"制御に関する動詞"を採り上げる．規則選択パターンは，必須格に主格，対象格を，自由格に時格を持つ．説明生成するとき，時格が時刻格ならば，概念的に分類した動詞は'開始する'と '終了する' の２つの場合が当てはまり一意に決定できない．この問題を解決するために，便宜的に格スロットの数を増やしておくことが考えられる．つまり，時点格が時点格１ならば動詞を'開始する'に，時点格２ならば動詞を'終了する'にする．しかし，格スロットを増やし過ぎると，動詞決定規則の数を抑えれなくなる．そこで，格スロットを増やす際には，動詞決定規則の数をできるだけ少なくするように留意する必要がある．

３．３　構造情報に基づく説明文の洗練化

構造情報に基づく説明文の洗練化とは，説明するメソッドの構造情報に記述されている下位メソッドのメッセージセンドの構造に基づいて機能情報から指示語，接続詞，省略を用いて柔軟な説明文を生成することをいう．以下，簡単に柔軟な説明を生成するための基本方針を述べる．

(1) 場所格が始点格ならば，動詞を'出る'，
　始点格に対応する助詞を'から'とする
(2) 場所格が終点格ならば，動詞を'到着する'，
　終点格に対応する助詞を'へ'とする．

図４　動詞決定規則

（１）文の関係に応じた接続詞を用いる．

連続する２つの文において，動詞の振舞いが相反するときには，逆接の接続詞'しかし'を，動詞の振舞いが異なるときには，順接の接続詞'そして'を，動詞の振舞いが同じならば，接続詞'同様に'を第２の文の先頭に付け加える．

（２）既出の事象に対して，指示語を用いる．

複数の機能情報において，対象格，方法格のうち同一の格情報が現れたときには，最初に現れた格情報を除いて，他の機能情報に現れる格情報を指示語で記述する．ただし，複数存在する場合は，指示語にしない．

（３）暗黙の内に明らかなことは省略する．

複数の機能情報に現れる格情報の中で，主格，場所格が同じ場合には，後続の機能情報の説明生成時にはそれらを省略する．

ユーザがメソッドの内容についての説明を要求するとき，システムは，手続きの構造情報に基づいて説明を生成するので，基本構造（逐次，条件分岐，反復）に応じた洗練化規則が必要となる．以下に各々の基本構造における洗練化規則の例について述べる．

（１）逐次構造における洗練化規則

[(イ)(ロ)(ハ)の条件がすべて真ならば，(ニ)を実行する．]

(イ)逐次構造である．

(ロ)機能情報内の動詞名が同じである．

(ハ)下位メソッドの機能情報の中に主格を除く格情報がすべて同じものがある．

(ニ)機能情報の格フレームにおいて主格を'と'で結び，機能情報の格フレームを１個の機能情報の格フレームにする．

（２）条件分岐構造における洗練化規則

[(イ)(ロ)(ハ)の条件がすべて真ならば，(ニ)を実行する．]

(イ)条件分岐構造である

(ロ)条件が真のときの実行文１がある．

(ハ)条件が偽のときの実行文２がある．

(ニ)テンプレート{もし'条件'が真であるならば，'実行文１'を実行する．そうでなければ，'実行文２'を実行する．}を利用し，'条件'，'実行文１'，'実行文２'の説明文を生成し，それらをテンプレート内に埋め込む．

（３）反復構造における洗練化規則

[(イ)(ロ)の条件が真ならば，(ハ)を実行する．]

(イ)反復構造である

(ロ)条件が真のとき実行文が実行される．

(ハ)テンプレート{'条件'が真である限り，'実行文'を繰り返す．}を利用し，'条件'，'実行文'の説明文を生成し，それらをテンプレート内に埋め込む．

４．説明生成機能の実現

図５に本システムにおける説明生成部の構成を示す．本説明生成部は，３つの処理モジュール（機能情報検索モジュール，構造情報検索モジュール，説明生成・洗練化モジュール）と５つのファイル（機能情報ベース，構造情報ベース，動詞決定規則ベース，洗練化規則ベース）から構成される．以下に各モジュールの機能を述べる．

①機能情報検索モジュール

まず，説明生成・洗練化モジュールより指定されたメソッドの機能情報を機能情報ベースから検索する．次に，その機能情報に対応する規則選択パターンと動詞決定規則を動詞決定規則ベースより検索し，それらを説明生成・洗練化モジュールに渡す．

②構造情報検索モジュール

説明生成・洗練化モジュールで指定されたメソッドの構造情報を構造情報ベースより検索し，それを説明生成・洗練化モジュールに渡す．

③説明生成・洗練化モジュール

本説明生成部の中心的な役割を担うモジュールで，ユーザの要求に従って説明を生成する．

４．１　説明文生成の手順

以下に，単一の機能情報を用いた説明生成過程と複数の機能情報を用いた説明生成過程における説明生成・洗練化モジュールの働きについて述べる．

１）単一の機能情報からの説明生成

ア）　説明生成・洗練化モジュールはユーザから説明を要求されたメッセージ名を機能情報検索モジュールに与え，格文法表現された機能情報と説明生成規則（動詞決定規則，規則選択パターン）を受け取る．

イ）　機能情報と規則選択パターンとを照合し，パターン内の格変数に，機能情報に現れる格情報を代入する．代入された格変数の組合せにより，動詞決定規則を用いて動詞を決定し説明文を生成する．

２）メソッドの構造情報とそのメソッドを構成する複数の下位メソッドの機能情報からの説明生成

ア）　説明生成・洗練化モジュールはユーザから説明を要求されたメッセージパターンを構造情報検索部に与え，そのメソッドを構成する下位メソッドのメッセージパターンとそのメソッドの構造を得る．

イ）下位メソッドのメッセージパターンを機能情報検索部に与え，それぞれ，機能情報と説明生成規則を得る．

ウ）機能情報に洗練化規則を用いて洗練化する．洗練化された機能情報と規則選択パターンを照合し，動詞決定規則と洗練化規則を用いて説明を生成する．

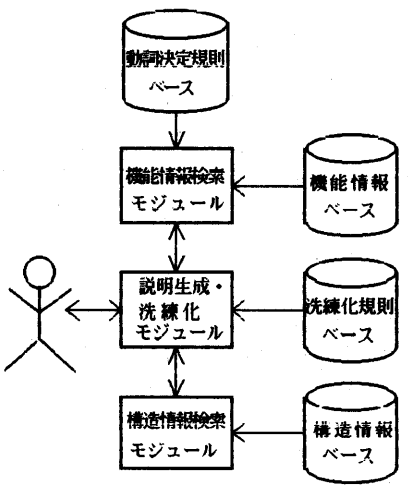

図５．説明生成部の構成

4．2　ドキュメント情報ベース

4．2．1　機能情報ベース

機能情報ベースには，３．２で述べたソフトウェアのメソッドの機能情報が格納される．機能情報の記述形式を図６に示す．ここで，actorは主格，objectは対象格，wayは方法格，fromは始点格（場所格），toは終点格（場所格），duringは時間格（時格），from2は開始格（時格），untilは終了格（時格），at1とat2は時点格（時格）を示す．また，(*var*格の名称)は，格に対応する格情報を意味する．また，動詞は概念的な意味で分類されており，verbにはそのカテゴリー名が入る．さらに，時点格はat1，at2に増やされている．また，メソッドが引数を持つ場合には，それぞれ引数が対応する格スロットは引数名をもつ．

説明文の洗練化を容易にするために，各格情報に修飾する語がある場合における記述形式の一例を図７に示す．

4．2．2　構造情報ベース

構造情報ベースには，メソッドの構造情報が格納される．現段階では，構造情報を以下の情報とし，ユーザにより記述する．

（１）メソッド内のメッセージパッシングの構造

　　現在，メソッドに許される構造は逐次，条件分岐，反復で，３レベル以上の入れ子構造を許さない．

（２）単一メソッドを構成している下位メソッド数

その記述形式を図８に示す．ここで，structureの値には逐次，条件分岐，反復構造のいずれかを，numberの値には以下に示すように，メソッドの構造情報の種類に応じて単一メソッドを構成している下位メソッド数を記述する．

（１）逐次構造

　　逐次に構成されている下位メソッド数をnとし，(number n)のように記述する．

（２）条件分岐構造

　　条件が真のとき，実行文を構成している下位メソッド数をm，偽のとき，実行文を構成している下位メソッド数を1とし，(number m 1)のように記述する．

（３）反復構造

　　条件が真（偽）のとき，実行される実行文を構成している下位メソッド数をmとし，(number m)のように記述する．

4．3　説明生成規則

本節では，動詞決定規則ベースと洗練化規則ベースの構成，及びそれらの記述形式について述べる．

```
(functional_information
     (message_pattern (メッセージパターン)
     (formal_argument (引数名))
               (verb ((actor (*var*act))
                      (object (*var*obj))
                      (way (*var*way))
                      (from (*var*from))
                      (to (*var*to))
                      (during (*var*dur))
                      (from2 (*var*from2))
                      (until (*var*utl))
                      (at1 (*var*at))
                      (at2 (*var*at1)))))))
```

図６　機能情報の記述形式

シミュレーションオブジェクトのタスク

(a) 修飾語句を持つ格情報

（タスク　（シミュレーションオブジェクト））

(b) 格情報の格フレーム内の記述形式

図７　格情報の記述形式の一例

```
(structural_information
      (message_pattern (メッセージパターン))
      (structure (メッセージパッシングの構造))
      (number (下位メソッド数１
              [下位メソッド数２
               下位メソッド数３ ]))
      (    (下位メソッド名1
              (formal_argument (引数)))
                    :
           (下位メソッド名n
              (formal_argument (引数)))  )
     [(    (下位メソッド名1
              (formal_argument (引数)))
                    :
           (下位メソッド名1[m]
              (formal_argument (引数)))  )] ))
```

図８　構造情報の記述形式

4．3．1　動詞決定規則ベース

動詞決定規則ベースは，概念的な意味により分類された動詞に対応する規則選択パターンと動詞決定規則を格納する．各々のパターンは，そのパターン特有の動詞決定規則に対応づけられている．一例としてplace_verbの規則選択パターン，及び動詞決定規則をそれぞれ図９，図１０に示す．place_verbは，格フレーム表現において，必須格にactor，at，wayを設定し，自由格であるfrom，toの種類によって，どの動詞決定規則を適用するかを決定する．

4．3．2　洗練化規則ベース

洗練化規則ベースは，逐次，条件分岐，反復などのメソッドの論理構造情報に応じ，説明を生成するためのテンプレートを格納する．

洗練化規則に対する記述例を以下に述べる．

（１）逐次構造

```
(if (and (structure '逐次構造)
         (equal (verb 機能情報1)
                (verb 機能情報2))
         (match_case 機能情報1 機能情報2))
 then (1) (replace_case 'actor
           (append 主格名1 '(と) 主格名2))
```

（２）条件分岐構造

```
(if (and (structure '条件分岐構造)
         (equal (length
                  (get_slot number)
                  3)))
  then (create_explanation
        'ifThenElse
         説明文1 (generate_exp 条件部))
         説明文2 (generate_exp 実行文1))
         説明文3 (generate_exp 実行文2))))
```

（３）反復構造

```
(if (and (structure '反復構造)
 then (create_explanation
       'whileTrue
       説明文1 (generate_exp 条件部)
       説明文2 (generate_exp 実行文))
```

4．4　動作例

ここでは，メソッドdefineArrivalScheduleの構造情報と下位メソッドの機能情報から説明について考える．このメソッドの定義を図１１に，下位メソッドである'scheduleArrivalOf: accordingTo: startingAt'のメソッドの機能情報記述を図１２に示す．

システムは，４．１節で述べた手順により次に示す説明文を生成しユーザに提示する．

```
(place_verb (actor (*var* act))
            (at (*var* at))
            (way (*var* way))
            {from (*var* from)}
            {to (*var* to)}))
```

ただし，( )内の格は必須格を，{ }内の格は自由格を示す．

図９　規則選択パターンの一例(place_verb)

```
(1) (if  (and (exist from) (not (exist to)))
    then (set_word from から place_verb 出る))
(2) (if  (and (not (exist from)) (exist to))
    then (set_word from へ place_verb 到着する))
```

図１０　動詞決定規則の一例(place_verb)

```
defineArrivalSchedule
  self scheduleArrivalOf:MainlandArrival
       accordingTo:(Exponential parameter:0.15)
       startingAt: 420.0
  self scheduleArrivalOf:IslandArrival
       accordingTo:(Exponential parameter:0.15)
       startingAt: 420.0
  self scheduleArrivalOf:(Ferry new)
       at: 420.0
```

図１１　メソッド'defineArrivalSchedule'の定義

```
(functional_information
  (message_pattern
    (scheduleArrivalOf:aSimulationObjectClass
     accordingTo:aProbabilityDistribution
     startingAt:timeInteger))
  (formal_argument (aSimulationObjectClass
                    aProbabilityDistribution
                    timeInteger)
        (place_verb
            (actor (インスタンス
                    ^aSimulationObjectClass))
            (at (指定時刻 ^timeInteger))
            (way (確率分布
                  ^aProbabilityDistribution))
                   (to (シミュレーション)))))
```

図１２　メソッド'scheduleArrialOf: accordingTo: startingAt'の機能情報記述

”インスタンスMainlandArrivalとインスタンスIslandArrivalは，指定時刻420.0に確率分布（平均 0.15の指数分布）にしたがって，シミュレーションへ到着する．同様にクラス Ferryのインスタンスも指定時刻420.0にシミュレーションへ到着する．”

5．むすび

オブジェクト指向型言語により記述された待ち行列シミュレーションソフトウェアを対象として，そのドキュメント情報を用いたソフトウェアの説明を生成する機能について考察した．本システムの特徴として以下のものが挙げられる．

（１）説明を生成するための情報としてソフトウェアのドキュメント情報（構造情報と機能情報）に着目し，ソフトウェアの機能をその構造に基づいてトップダウン的に説明する機構を実現した．本説明生成機構はソフトウェアが階層的に記述されている場合に，あるメソッドをその構造情報に基づいて下位メソッドに分解し，説明を生成する．したがって，ユーザの要求に応じて，ユーザが理解できるメソッドあるいは既知のメソッドまで分解して説明を生成することができるために，ユーザの対象ソフトウェアに対する理解レベルに対応した説明ができると考えられる．

（２）柔軟な説明を生成するために，機能情報の意味表現形式として格文法を採用し，意味表現から説明文を生成する機構を実現した．さらに，複数の意味表現を組み合わせて，説明文を生成するために，ソフトウェアの構造情報（逐次，条件分岐，反復構造）に応じた洗練化規則を用いて，柔軟な説明を生成することを提案した．その結果，プログラマのソフトウェアに関する把握が容易になると考えられる．

（３）ソフトウェアの構造に基づいて説明が生成されるために，この説明を理解することにより，ユーザがソースコードの論理的構造を把握するための手助けになると考えられる．

最後に，今後の課題として以下のものが挙げられる．

（１）現段階では，説明の生成が可能な構造情報を逐次，条件分岐，反復構造に限定し，複雑な入れ子構造を許していない．そこで，ソースコードを構文解析し，構造情報を自動的に抽出し，複雑な入れ子構造に対応できるように構造情報検索部を構築する必要がある．

（２）より柔軟な説明文を生成するために，語の省略規則，あるいは，指示語・代名詞の適切な置き換え等，洗練化規則を抽出する必要がある．

（３）現在，機能情報はソフトウェア開発者により記述されているが，これはソフトウェアの意味に相当するものと考えられ，将来的には計算機による抽出を検討したい．

謝　辞

日頃，御指導頂く愛媛大教授高松雄三先生，同教授相原恒博先生に感謝の意を表する．又，熱心に御討論頂いた阪大産研山口高平助手，手塚研の諸氏に厚くお礼申し上げます．

［参考文献］

[1] 山田，山口，真田，角所，手塚：”ソフトウェアの変更におけるユーザインタフェースの高度化を目指す記述言語”，信学論（Ｄ），J70-D，No.11，pp. 2308-2313 (1987).

[2] 山田，手塚：”ドキュメンテーションを利用したソフトウェア説明機能”，情報処理学会 第３５回全国大会，5N-1 (1987).

[3] Goldberg, A. and Robinson, D. : "Smalltalk-80 The Language and Its Implementation", Addison Wesley (1983).

[4] Swartout, W.R. : "XPLAIN: a System for Creating and Explaining Expert Consulting Programs", Artificial Intelligence, Vol. 21, No.3, pp.285-325 (1983).

[5] Swartout, B. : "GIST English Generator", AAAI-83, pp.404-409 (1983).

[6] 難波，安西，中島：”考える道具としてのＬisp入門” 共立出版 (1986)