ソフトウェア工学　136－27
（２００２．３．８）

保守過程における自然言語を用いた要求記述支援システムの検討

滝沢陽三*　　上田賀一**

*茨城高専　　**茨城大学

ソフトウェア保守過程における自然言語記述からの要求導出の手法および支援システムについて提案する．保守過程における要求の種類を考察し，再設計・再構築を必要とする既存システムに対して顧客が自然言語を用いてどのような記述を行うかを議論する．本研究で実現を進めている要求仕様導出支援システムに対して修正・追加を行うことで，これらに対応する機能の実現を試みるとともに，問題点や課題を示す．

Supporting System for Requirement Analysis in Maintenance Process using Natural Language

Yozo Takizawa*　　Yoshikazu Ueda**

*Ibaraki National College of Technology
**Ibaraki University

This paper describes the idea of acquiring requirements from natural language descriptions in the software maintenance process. Firstly author defines several kinds of requirements in the maintenance that includes parameter setting and re-design. Secondly the system which author has designed and structured is described, it supports to acquire requirement descriptions written in natural language from informal descriptions. The extentions for the system are suggested to support customer's requirements on the maintenance process.

1. はじめに

コンピュータソフトウェアは年々巨大化・複雑化し, コンピュータが果たす役割そのものを実現する主体であると言っても過言ではなくなった. 利用者はプリミティブな演算・制御機能ではなく, 日常的な道具としての総合的な機能を求めるようになっている. ネットワーク基盤整備の進行がその傾向を更に強め, コミュニケーションツールとしての役割も果たしている.

この情勢において, 人々のソフトウェアに対する要求は複雑で多様化しているが, 同時に, 既存のソフトウェアに対する不満点の解消や機能の追加といった形の要求も数多く存在する. ワードプロセッサや表計算等, ソフトウェアの種類が各利用分野において固定されつつある現在においては, むしろそういった変更要求の方が要求全体の多くを占めている. したがって, 変更要求を迅速に取り入れるソフトウェア開発手法の発展が必須である. これは, 単純に既存ソフトウェアの変更・追加技法だけではうまくいかず, 新規開発において変更要求を受け入れやすいようあらかじめ設計しておく必要がある. ソフトウェアのライブラリ化やオブジェクト化といった再利用の側面だけでなく, システムとしての同一性を確保しつつ柔軟な変更・追加が可能でなければならない.

筆者らは, 自然言語で記述された要求記述についての研究を行ってきたが, これまでは主に新規ソフトウェアの開発に焦点を当ててきた. これは, 本研究の中心が顧客によるソフトウェア要求定義を可能にする手法およびシステムの確立だからである(図 1). しかし, 昨今の要求定義は要求モデルをベースとしたものが主流であり, 既存のソフトウェアシステムを可能な限り再利用可能にするとともに, 保守過程においても容易に変更が可能であるようにする傾向が高い. これは, 特に大規模システムの構築に見られ, Java をはじめとしたモジュール性の高い言語システムを採用してソフトウェア部品をデータベース化し, 顧客の要求に応じた開発・運用を可能とする汎用システムの発達による. このような汎用システムの導入によって開発・運用コストは下がり, 状況に応じたシステム変更も柔軟に対応できることになる. しかしそれだけに, 顧客からの要求を的確かつ詳細に把握することが必要となり, UML を代表とする視覚的な要求定義を含む開発過程が主流となっている.

本稿では, 保守過程におけるこのような機能・システム変更要求に対して自然言語記述が果たす役割について考察を行い, そのプロセス化および支援システムの実現について述べる.

2. 既存ソフトウェアに対する要求

既存システムに対する要求の多くは, 利用者(顧客)が使用・運用した結果判明した問題点や改良点についてのものである. これはソフトウェア一般にも言えることだが, 近年のソフトウェアはソースコード等が膨大になっており, ハードウェアと同様に設計書や仕様書の段階まで遡って修正や改良を行う必要がある. しかし,

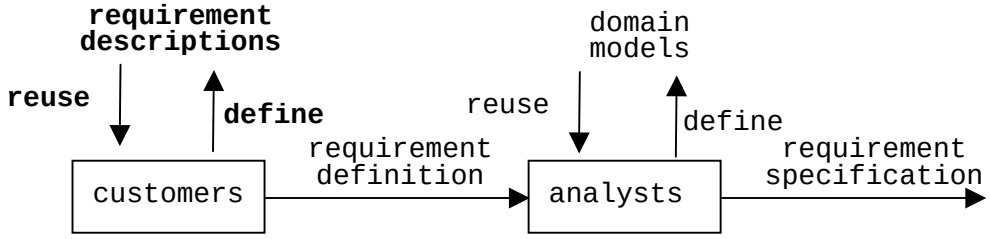

図1 顧客によるソフトウェア要求定義の流れ

昨今のコンピュータシステムはソフトウェアの変更のみで機能そのものを変更できることから導入が急速に進んでいるのも事実であり，変更期間を含むコストはハードウェアの変更よりも低く押さえられるべきである．このためには，仕様化・設計の時点で要求に対する柔軟な対応を想定することになる．

既存ソフトウェアに対する要求としては以下のようなレベルが考えられる．

・設定値の変更で対応できるもの
・データベースの変更で対応できるもの
・ソフトウェアモジュールの交換で対応できるもの
・システムの再設計が必要なもの

以上はあくまで一例だが，実際にはこれらの要求が混在したものとなり，開発者は複数のレベルでの変更・修正を行うことになる．この場合，要求を出す顧客側はどのレベルであるかということは通常意識せず，開発者が判断して対応する．再設計が必要なものは新規開発の場合も考えられ，顧客にとっては別のシステムとの置き換えに相当する可能性もある．

顧客が開発技術に精通しておらず，契約上の知識・技術のみ修得している場合，要求は自然言語で記述もしくは口頭で開発者に伝えられることが多いだろう．しかし，設定値やデータの変更で対応できるのであれば，顧客側がシステムを維持しつつ変更できる可能性が高い．もしそうであれば顧客側で早急な対応ができ，コストは大幅に削減できる．無論，開発側と顧客側の間には責任分解点があるが，コンピュータシステムの普及が広がっており，顧客側がシステム変更の一部に責任をもつ例は増えていくだろう．要求を表現する一般的な手段としての自然言語が，保守過程における要求の反映に用いることができるのであれば極めて有効である．

3．要求記述の形式と自然言語

現在の開発技術においては，要求記述は様々なドキュメントで構成されている．顧客側と開発者側双方で理解可能な文書も存在するが，多くは設計過程につながる開発者のみ理解するべき文書である．このような文書であっても，昨今では UML 等を用いて形式化し，他の開発者も的確に用いることができるという観点では再利用の可能性は高い．しかし，顧客が責任を担える範囲の文書ではなく，顧客自身が保守過程において利用できるものではない．ただし，図式化されているものが多いため，顧客が視覚的に確認を行う場合もある．

UML のようなモデリング言語を用いた開発方法論では，自然言語は要求記述を行うには不完全な手段であるとし，概要説明を行うための手段に留まっている．自然言語は簡単な概念が人により解釈がまちまちになるだけでなく，一つの概念について様々な表現が可能なため繁雑になるからである．また，その自然言語も極めて形式的な制約が設けられており，専用の文法に従った単語レベルでの利用に限定されていることもある．

筆者がこのような自然言語を要求記述の手段の中心に据えているのは，自然言語が顧客/開発者問わず全ての人間が利用できる手段であるということ，初期の要求形成に少なからず自然言語が大きな役割を果たしていることからである．そこには，開発技法の知識をもたない顧客が関与できる余地が十分に残されている．本研究では，以下のような処理手続きから構成される，自然言語記述をベースとした要求導出支援システム(ARDES, the System for Acquiring Requirement and Deriving Specification from Descriptions written in Natural Language)を考案し開発を試みている．

・字句解析部 -- 顧客から非形式的な要求記述を受け取り単語を認識する．

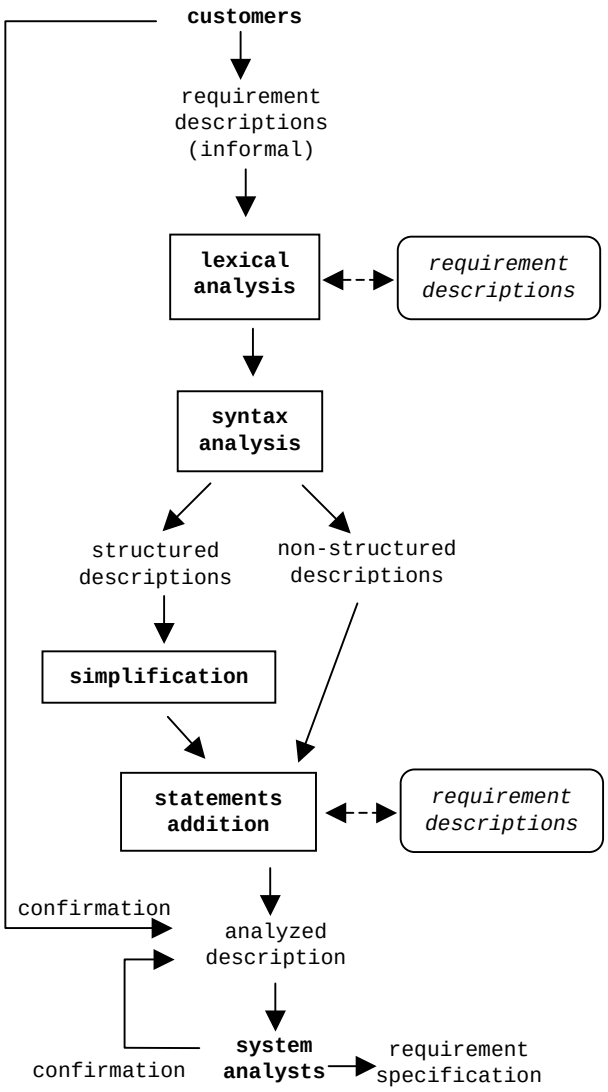

図 2 支援手法 ASRED の流れ

・構文解析部 -- 要求記述としての構文をもつ部分を抽出する.
・簡略化部 -- 要求記述を単文化し, 図式表現と対応付ける.
・文章追加部 -- 認識された単語を用いて既存の関連する要求記述を追加する.

これらは実際には順番に処理されるとは限らず, 文章追加後の形式化された要求記述が顧客によって確認され, 更に修正が加えられることも想定される.

このシステムの特徴は, 顧客が用いるシステムであるということである. システムの維持には開発技術にある程度精通した者を前提とするが, 顧客はこのシステムを用いて自然言語記述を要求に従って形式化していくとともに, 形式化に直接携わることによってその分野の形式的表現を学ぶ機会が得られる.

このような手法を本研究では ASRED (the Acquirement Supporting Method of Requirement from Descriptions)としてまとめ, 支援システム ARDES の運用基盤として定義している(図 2).

4. 保守過程における支援システムの適用

前章で挙げた支援システムは, 新規のソフトウェア開発を想定している. 既存の記述を用いてはいるが, あくまで導出のための参考データという位置付けである. 本章では, このシステムを保守段階にある変更が必要なソフトウェアに対して用いる場合を想定し, どのような処理がなされることになるかを考える.

与えられる要求記述は, 顧客にも読解できる自然言語記述であるが, 開発されたソフトウェアシステムのものであるため既に形式化されている. この記述に顧客が変更要求を自然言語で加えてシステムに与えると, 4 種類の記述が導かれる可能性がある.

(1) 既存の形式的記述
(2) 修正が施された既存の形式的記述が, 支援システムにより再形式化された記述
(3) 新規の非形式的記述が, 支援システムにより形式化された記述
(4) 新規の記述・修正によって新たに関連付けられた, 既存の形式化された要求記述

これら以外にも, 修正が施されたことによって形式化の中で消滅した記述が存在する可能性もある. ここでは, 既存の要求記述が充実していることを前提として, 必要な情報をもつ記述

は消滅しないこととする(すなわち，必要がなくなった記述と認識する)．4 種類の記述について，(1)はそのまま残ったもの，(3)は新規開発と同じ導出である．(2)および(4)が，変更要求によって新たに得られる記述である．(2)はまさしく機能・システム変更によって変化した記述であり，モジュールの変更を必要とする要求と言える．(4)はこれまでになかった機能・システムを加えることを示す記述であり，新規モジュールの追加，もしくは外部の別のシステムとの連動を新たに示す記述であることが想定される．

5．要求記述導出支援システムの拡張

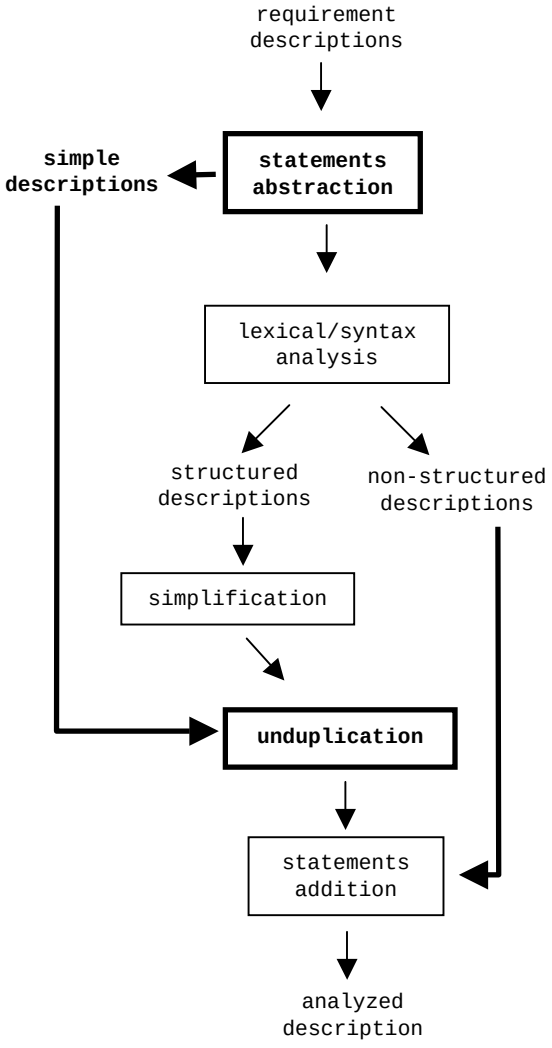

図 3 拡張された ASRED (抜粋)

5.1 処理手順の追加

先に示した 4 種類の記述のうち，(2)(4)は現在の支援システムでは想定されていない．そのままの構成でも過不足なく導出される可能性はあるが，保守過程で効率よく用いるにはそのような記述を前提としたシステムにする必要があると思われる．ここでは，以下のような処理手順を加えることにする．

・ 形式記述の抽出(statements abstraction)
システムで想定されている形式性を有した記述を抽出する．字句・構文解析部の前に位置し，単文化部で参照されている構文規則を用いて判定する．また，新規システムの記述に対しても適用することとし，顧客が当初から形式的記述を行った場合は後の解析・単文化が省略される．逆に，保守過程において既存記述に変更を行ったにも関わらず，想定されている形式性を保っていた場合もこの段階で抽出される．

・ 類似の記述の重複解消(unduplication)
内容的に同じと判断される記述を統一する．簡略化部の後に位置し，上記の抽出された形式記述を融合する．文章追加部の前に行うのは，前述の抽出部において変更・追加部分がそのまま抽出される可能性があり，その場合は新規記述と同様に，単語を用いて関連する記述を追加しなければならないからである．なお，後の文章追加部でも同じ単文が導出され解消される処理があるが，これは辞書情報を追加する際の処理として行われるため区別している．

以上の処理手順を追加した拡張版 ASRED の概要を図 3 に示す．

5.2 具体例

以下では，解析後の単純な要求記述文(英語)を用いて処理手順を解説する．

まず，新規の非形式的記述として以下の文章が与えられたとする．

・An automatic teller machine is connected to the main computer which inquires of an appropriate bank.

支援システムは,新規記述として以下のような形式記述を導出する[1].

・automatic teller machine is connected to main computer
・main computer inquire of branch office
・ATM is automatic teller machine
・main computer is computer
・branch office is office

上記導出はあくまで例であり,初期の辞書情報や顧客・開発者の判断によって大きく変わる.

ここで,上記導出文章に対して追加・変更要求があり以下のように書き換えられたとする（下線部は追加・変更部分).

・automatic teller machine is connected to central computer which is located in the head office
・central computer inquire of branch office
・ATM is automatic teller machine
・central computer is computer
・branch office is office
・CD has the subset of the ATM functions.

従来の記述において``main computer''とされていたものが``central computer''に変更され,CD（キャッシュディスペンサ）に関する記述が追加されている.

ここで,新規に追加された手順である形式記述の抽出が行われると,2～4行目は構文面では形式化されていると判断できるため取り除かれる（2行目と4行目は変更が加えられているにも関わらず抽出されることに注意).抽出記述は後に融合されるとして,残った記述は以下の2行である.

・automatic teller machine is connected to central computer which is located in the head office
・CD has the subset of the ATM functions.

この2行に,新規記述として解析を行うと以下のようになる(英語記述の場合は原則として名詞および動詞（相当）のみで構成される単文に変換される).

・automatic teller machine is connected to central computer
・central computer is located in head office
・CD is ATM

3行目は,字句・構文解析部において冠詞等が除かれた後に,単文化部において``*A* has/have subset of *B* function''が常套句として``*A* is *B*''に相当すると判断した結果得られたものである(システムが要求記述解析に特化されていることを前提としている).更に,字句解析部において``central computer''は一つの語として扱われるため,元の記述は``main computer''と置き換わったことになる.

この後,前もって抽出された記述が融合される（この例では重複は存在しない).

・automatic teller machine is connected to central computer
・central computer is located in head office
・CD is ATM
・central computer inquire of branch office
・ATM is automatic teller machine
・central computer is computer
・branch office is office

文章追加部では,新規に登場した``central computer''および``CD''について辞書情報が

検索され，それぞれ以下の 3 つが得られるものとする．

・central computer is computer
・CD is cash dispenser
・cash dispenser is automatic teller machine

最終的には以下の記述が結果としてシステムより導出される．

・automatic teller machine is connected to central computer
・central computer is located in head office
・CD is ATM
・central computer inquire of branch office
・ATM is automatic teller machine
・central computer is computer
・branch office is office
・central computer is computer
・cash dispenser is automatic teller machine

この後，顧客自身によって内容の検討が行われ，不適切と思われる単文があれば（解析後の単文群ではなく）元の記述を修正する．

先にも述べたように，導出結果は辞書情報および顧客・開発者の判断によって大きく変わる．たとえば，``CD''(cash dispenser)という情報を加えたにも関わらず辞書情報として関連情報がなかった場合は``CD is cash dispenser''が追加されず，その後顧客によって，CD に関しては ATM に統合するものと解釈されるかもしれない．一方，ATM と CD を明確に区別したいと解釈した場合は，辞書情報に関わらず顧客が``CD is cash dispenser''に相当する記述を別途追加し，それが最終的な要求記述として導出される可能性もある．

5.3 支援システムの実装

本研究で試作を進めているシステム ARDES は，自然言語を扱う処理が多いため，プログラミング言語として LISP 系言語の Scheme を用いている．実際には，グラフィカルなユーザインタフェースを提供するため，ツールキットを備えた STk を用いている．図 4 は，X Window System 上での実現例である．

6．課題

記述の融合部については，基本的には同一文章の重複を解消するが，既存の記述に対し用法が異なっただけの表現が追加・導出されることも考えられる．このため，内容的に同じと判断するために，既存の辞書情報に付加的な情報を追加し利用することが考えられる．

形式記述の抽出および融合処理において，導出される記述の種類の違いによって，どのレベルの要求かを半自動的に認識できる可能性が高い．設定値やデータベース情報の変更で対応できるものについては記述では現れにくいが，モジュールの交換やシステムの再設計が明らかに必要であることは，処理の途中で，もしくは追加・変更前の記述と比較して導くことができることが考えられる（新規記述の解析等）．顧客が開発者に依存せず認識できるのであれば極めて有効であるが，この機能を実現するためには処理部や辞書情報に大幅な修正・追加を行う必要がある．

システムの実装は Scheme 処理系を用いているが，要求記述や辞書情報の規模が大きくなるほど処理が遅くなる．インタプリタ上で必要に応じてメモリを再帰的に確保していくためだが，本研究ではユーザインタフェースについても Scheme 処理系で実現していることもあり，実用面で大きな問題点となっている．このため，辞書情報の検索を中心に記述処理の一部を分離して C 言語で書き直すとともに，ユーザインタフェースを HTTP 経由にすることを試みている．この場合，記述内容を送信フォームもしくはファイル指定で行うとともに，記述

図 4　ARDES における支援ツール

処理や処理結果の表示を CGI で実現する必要がある．自然言語処理については，本システムが試作段階ということもあり LISP 処理系を連動した形を想定しているが，現段階では Scheme 処理系を CGI として使用する際には制約が多く，システムを実現するためには処理系を拡張する必要があると思われる．

参考文献

[1] 滝沢陽三，上田賀一: 顧客による要求分析のための手法とその支援システム，情処研報，Vol.2000, No.25(2000).

[2] 滝沢陽三，上田賀一: 要求仕様導出支援システムにおける辞書構築手法，情処研報，Vol.98, No.64(1998).

[3] 滝沢陽三，上田賀一: 自然言語記述による要求仕様導出支援システムの提案，情報処理学会論文誌，Vol.38, No.3 (1997).

[4] E.Amoroso: Creating Formal Specifications from Informal Requirements Documents, ACM SIGSOFT SEN, Vol.20, No.1 (1995).

[5] 大野雅志，原田 実: オブジェクト指向分析支援システム CAMEO -日本語文章記述からの設計要素の自動抽出-，情処研報，Vol.SE99-14(1994).

[6] 磯田定宏，黒木宏明: 統合化 CASE システム SoftDA の機能，コンピュータソフトウェア，Vol.10, No.2(1993).