ソフトウェア工学 126－3
（2000．3．9）

顧客による要求分析のための手法とその支援システム

滝沢陽三*　　上田賀一**

*茨城高専　　**茨城大学

要求の記述・定義を自然言語によって顧客自身が行うための方法論と，その支援システムについて報告する．通常，顧客自身は開発手法の中で支援する対象ではないが，顧客側が実際のソフトウェア開発を意識して要求を定義できれば要求の検証が行いやすくなるだけでなく，開発者側による要求の理解も高まる．本研究では，顧客が要求を記述する手段として自然言語を想定し，具体的な手順を定義すると共に，現在の自然言語処理技術を前提とした支援システム ARDES の開発を試みた．本稿では，顧客が手法と支援システムに対しどのように関わっていくべきかを議論し，システムの詳細を解説すると共に，実際の解析を通して問題点を検証する．

Supporting System for Requirement Analysis by Customer

Yozo Takizawa*　　Yoshikazu Ueda**

*Ibaraki National College of Technology
**Ibaraki University

Authors make a suggestion to establish a methodology of requirement definition by customer. Generally, customers do not write documents referred and supported at the software development. Requirement descriptions will be easy for customers to read and inspect if they can write the documents using natural languages. Authors present a supporting system, ARDES, based on the methodology and report issues derived from the applying examples.

1. はじめに

近年，要求定義・分析段階で既存要求の再利用を図ることが盛んであり，ドメインモデルをフレームワークやパターンといった観点で定義することが試みられている．この目的は，無駄なソフトウェア開発を避け，ソフトウェア要求に対する即応性と柔軟性を高めることである．

通常，要求分析を行う者は顧客にインタビュー等を試み，これまでの開発経験や既存の要求記述を含むドキュメントを基に分析を行ってソフトウェア要求を記述し仕様化する．しかし，顧客の要求はインタビュー等で過不足なく得られることを前提としており，要求分析は顧客ではなく設計・開発者のための仕様を記述する(図 1)．このため，要求分析によって作成された要求仕様を顧客が確認するのは一般に困難である．ユースケース図等で表現された分析結果を顧客に示すこともあるが，検証に必要な資料として渡されることは稀である．

もし顧客が自然言語によって自らの要求を十分に記述することができれば，開発者はその要求記述を基に仕様を定義しソフトウェアを実装できるはずである．しかし，現在自然言語で記述されるドキュメントとしてのシナリオ等は，開発者が記述する補足的なものであり，顧客の要求をそのまま表現できるものではない．

本稿では，要求の洗練化/モデル化を自然言語記述によって顧客自身が行うための方法論と，その支援システムについて報告する．

2. ソフトウェア要求定義と顧客の役割

多くのソフトウェア開発手法において，顧客はインタビューの対象であり，開発者が要求を分析するための資料を作成するのに必要な情報を提供する存在である．このため，顧客は一般に開発過程の外部に位置し，開発手法の中で支援する対象ではない．これは，顧客が提供する要求が過不足のないものであることを前提とし，ソフトウェア開発手法とは別の技法で支援するべきものであると捉えられているからである．

しかし，昨今のソフトウェアの大規模化と多様性から，単純に必要な機能を把握してそれを満たせば良いというものではなくなりつつある．操作性を含む使い心地といった面だけでなく，ネットワーク化に伴う複合化と需要の多面性を，保守過程においても滞りなく，コストパフォーマンスの側面も考慮しながら満たす必要がある．無論，これら全てがソフトウェア開発にそのまま反映されるわけではないが，ソフトウェアが提供する機能を明確にするという観点において，実際の設計において無視できる要素はむしろ少ないと言える．

また，大規模ソフトウェアを開発する過程においても，汎用ではない，特殊なライブラリや開発ツールを必要とすることがある．この場合は顧客もソフトウェア開発者であることが多いが，ライブラリやツールの開発に詳しいとは限らず，詳細かつ複合した要求を開発者側が把握し文書化しなければならなくなる．しかし，ライブラリやツールが具体的にどのように使われるかを，開発側が的確に想定し必要と思われるソフトウェアを必要なだけ作り上げるということはまず不可能である．

本研究ではこういった問題を解決するため，顧

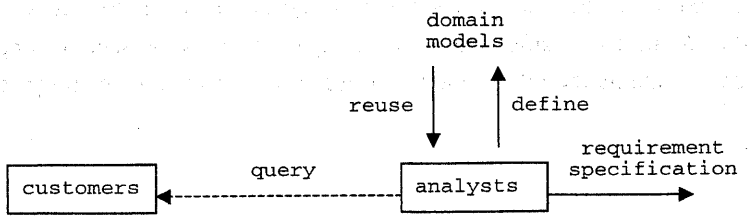

図 1 主要な開発手法におけるソフトウェア要求定義

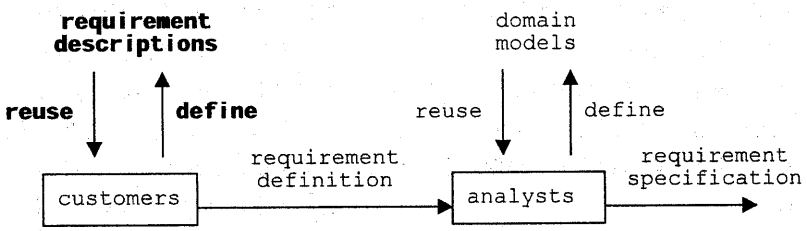

図2 顧客によるソフトウェア要求定義の流れ

客側が実際のソフトウェア開発を意識して要求を定義できる手法およびその支援システムの開発を試みている．開発に携わらない顧客が実際のソフトウェア開発の内容を前提とすることはできないが，その分野の既存ソフトウェアの成果がドキュメントとして残され整理されているのであれば，それを参照することで，実現可能性の検討も含めてより的確で詳細な要求として洗練できるはずである(図 2)．このようなドキュメントの記述方法としては，近年統一記法として広まりつつある UML があるが，UML は原則として開発者のためのものであり，その分野を熟知した開発者同士の密接な意思疎通を支援する目的として利用されるのが一般的である．このような観点から，本研究では要求記述の可読性を最優先とし，自然言語記述をドキュメントの記法としている．コンピュータによる支援システムを実現する場合，用いる自然言語は形式性を有していることが望ましいが，形式化しすぎると記述能力が低くなり，可読性も低下する．そこで本研究では，最初に要求を記述する際には非形式としつつ，支援環境によって形式化していく手順をとる．

3. 要求定義のプロセスモデル

前章の考え方に基づき，本研究では顧客による要求定義の支援手法 ASRED (the Acquirement Supporting Method of Requirement from Descriptions)を定義した．このプロセスモデルの目的は，顧客自身が形式言語や実装(プログラミング)技術を学ぶことなく，自然言語によって要求を定義できることである(図 3)．

ASRED は以下の 3 つのサブプロセスより成り立つ．

- 字句解析・構文解析：顧客によって記述された非形式記述から単語(word)および用語(phrase)を抽出する．単語および用語は構造化されるが形式化されない．
- 単文化：記述を単文化する．主に名詞と動詞のみによって構成される形式にする．これら

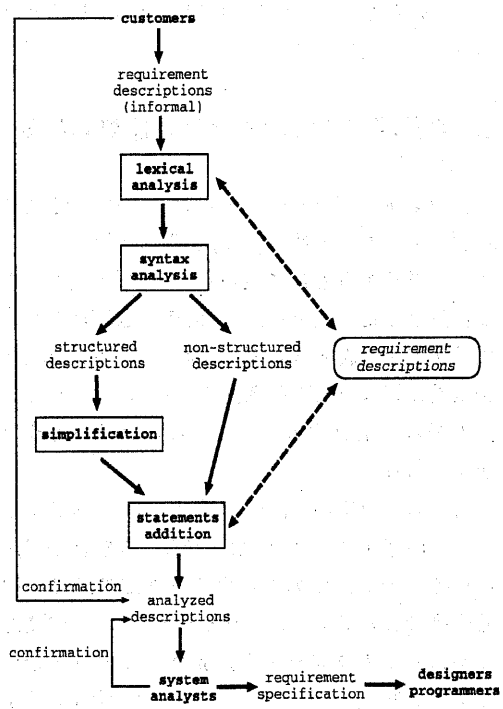

図 3 支援手法 ASRED の流れ

は極めて形式化されているが顧客によって理解することができる.

- 記述追加：既存システムの要求記述から関連する記述を検索し追加する. これらの記述も既に形式化され, 顧客によって理解できるものであるとする.

これらのプロセスにおいて, 自然言語処理は仕様要素抽出のための手段としてよりもむしろ, 顧客と開発者を結ぶインタフェースの役割を果たす. 顧客は要求を記述し, ASRED にしたがって処理を行い, 結果に従って記述を書き直していく. 最終的な記述は開発者によって設計・実装のための文書を記述するのに用いられる.

ASRED において最も重要な要素は, 既存システムの要求記述のデータベースである. 字句解析・構文解析は単語辞書を必要とし, 記述追加は単文化された既存の要求記述を必要とする. データベースには開発ソフトウェアのドメインではなく, 顧客の分野の単語・用語によって構成される.

4. 開発環境における自然言語処理技術

要求記述のデータベースおよびその処理は ASRED における要であり, 実際の支援システムとして構築する場合において, コンピュータによる自然言語処理技術が, より的確で過不足のない要求記述を定義するカギとなる. しかし, 自然言語処理技術は一般に不完全である. コンピュータが人間の言葉を的確に理解することは不可能だからである.

自然言語を用いた開発支援ツール・システムはいくつか存在する[3][4][5]. 特に, 自然言語記述から仕様の構成要素を抽出する機能は既存の開発環境においてよく見られるものであり, 開発方法論にもそのプロセスが定義されている場合がある. しかし, これらのツール・方法論はあくまで開発者によって用いられるものであり, 利用するにはソフトウェア開発特有の用語や技術等を身につけていなければならない. これは, コンピュータによる自然言語処理が不完全である以上, 人間による補助を必要とするからである.

もしコンピュータが自然言語記述をそのまま理解することができ, プログラミング言語として実行することができるのであれば, 開発者は実現可能性等を確認することができるだけでなく, ソフトウェア開発の自動化にも貢献する. しかし, 現在の自然言語処理技術は要求記述をプログラムとして実行するのに十分な機能をもっておらず, 極めて形式化された言語を理解するに留まっている.

しかし, 開発に必要な要求を記述することを支援するだけであれば, コンピュータは非形式な自然言語を理解する必要はない. 顧客自らが記述した文章について, それらが設計・実装において有用であるかどうかが判別できれば良く, 既存の要求記述を基に記述の抜けや不正確さを調べる支援を行うことはできる. 本研究ではこの考え方に基づき, 不完全な自然言語処理技術を前提とした支援システムの構築を目指している.

5. 支援システム

これまで述べた手法(ASRED)および自然言語処理技術の現状を踏まえて設計された支援システム ARDES (the System for Acquiring Requirement and Deriving Specification from Descriptions written in Natural Language)について述べる. 本研究ではこれまで形式的な日本語記述を中心にシステムを構築してきたが[1][2], 顧客による利用を重視した辞書情報の強化と, 日本語/英語混在といった多言語化を図っている. 本稿では支援システムについて, これらの拡張要素を含めて全体を解説する.

5.1 システム概要

ASRED に従って設けられる字句・構文解析部と単語・用語辞書検索部に加え, 特定の自然言語

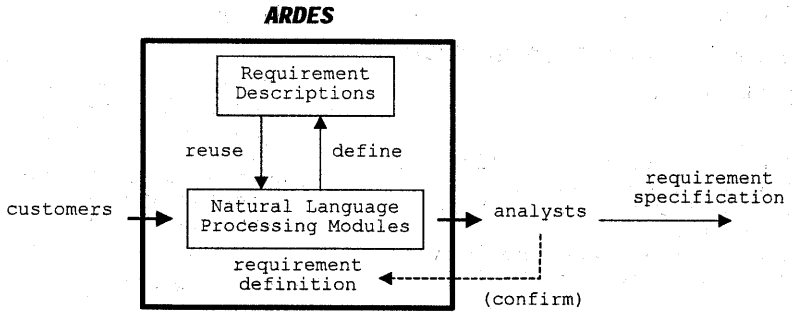

図 4 支援システム ARDES の概観

に依存した前処理部とユーザインタフェースによって構成される(図 4).

前処理部は,自然言語記述における句読点を含む記号について処理を行う. たとえば,自然言語として英語を用いる場合は,"$"という記号は"dollar"という単語に置き換えられる. 置き換えによって顧客による可読が困難になってはならず,また,記述における意味的変化を伴ってはならない.

ARDES システムには概念抽出部も含まれる. これは,最終結果から(顧客ではなく)開発者によって設計・実装に沿った仕様記述を導くのに必要な体裁に変換する. 概念抽出部は後の開発段階に必要な記述を得るためだけでなく,開発者によって検証され顧客に注意を促すためにも用いられる.

ARDES システムには更に単語・用語抽出部も設けられる. これらは既存の要求記述から必要な単語用語を抽出する役割があるが,実際は顧客が要求記述を洗練化するプロセスの一部として実現される. これは,現時点の要求記述を洗練化する過程で辞書情報も洗練化することを意味する.

5.2 具体的な処理手順

以下では,単純な要求記述文(英語)を用いて処理手順を解説する.

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

上記の要求記述を字句解析部によって処理を行うと,以下の単語(列)が出力される.

```
automatic teller machine, is connected to, main computer, inquire of, bank.
```

ここでは,単語辞書を用いて認識された単語のみが結果として残る. ただし,抽出されなかった単語を含む記述全体が単語・用語抽出部に渡され,辞書情報の洗練化に用いられる.

構文解析後の記述は以下のようになる.

```
automatic teller machine is connected to main computer which inquire of appropriate bank
```

この記述は,字句解析部で認識された単語に文章として必要な単語を加えた上で文章として認識されたものである.

次に,単文化処理が行われる.

```
automatic teller machine is connected to main computer
main computer inquire of bank
bank is appropriate
```

原則として名詞および動詞(相当)のみで構成される単文である. いくつかの単文は要求として不必要なものとなるが,この後の記述追加で使用す

るため残される.

記述追加部でいくつかの単文が追加された結果は以下のようになる.

```
automatic teller machine is
connected to main computer
main computer inquire of bank
bank is appropriate
automatic teller machine is ATM
main computer is computer
```

ここまではある程度手順に沿って解析・修正等が行われてきたが, 記述追加処理が行われた後の記述は, 顧客自身によって内容の検討が行われる. もしここで適切ではない単文があれば, 顧客は(解析後の単文群ではなく)元の記述を修正し, 再び解析が行われる. 解析後の記述は開発者によっても検討され, 既存の適切な記述によって辞書情報を洗練化していく.

たとえば, 顧客が上記の "bank is appropriate"を適切な記述ではないと判断し, 以下のように書き直したとする.

```
An automatic teller machine is
connected to the main computer which
inquires of the one of branch
offices.
```

この記述を再度字句解析した結果として

```
automatic teller machine, is
connected to, main computer, inquire
of, branch office.
```

が出力され, 構文解析した結果として

```
automatic teller machine is
connected to main computer which
inquire of branch office
```

が出力された場合, 単文化後の記述は以下のようになる.

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

記述追加後は

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

となることが考えられる.

以上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記述は大きく変わる. しかし, 要求記述そのものの判断・変更は顧客によってのみ行われ, 開発者は解析後の検証と辞書情報の洗練にのみ携わる. もちろん, 最終的な記述として出力された解析後の記述を, 後の開発に用いるために変更を行うことはあり得る. ただしその場合においても, 内容的な変更は(その記述自身を用いて)顧客に確認を求めることとなる.

5.3 実装例と辞書情報

本研究では試作システムを構築しているが, 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. 実際の処理系としては, グラフィカルユーザインタフェースを構築するためのツールキットを備えた STk を用いている. Scheme 処理系を用いる欠点としては, インタプリタ上で必要に応じてメモリを再帰的に割り当てていくため, 記述や辞書の規模が大きくなるほど処理が遅くなること, 状況によっては処理が不可能になる場合もあることである. 特に, 本研究で構築を試みているシステムは辞書情報の充実が重要であるため, これらは実用面で大きな問題点となる. このため, 記述解析部をより単純にした上で, yacc/lex 等を用いた C 言語処理系による実装を試みてはいるが, これらは本来プログラミング言語を実装するためのものとして作られていることもあり, 辞書情報の構築等において柔軟性に欠ける. 実用的な自然言語処理システムや WWW で見られる全文検索システムを併用する方法も考えられる

が，本研究で必要とされる自然言語処理はそれほど正確さを必要としないため，顧客が利用するには煩雑になるなどオーバースペックによる運用面での問題を生む可能性も考えられる．

Scheme による実装において，辞書情報は以下のように定義される．

[単語辞書]

```
(define word-dict-txt
   '((単語 @品詞) ...))
```

品詞については，自然言語(日本語，英語)にほぼ共通の名詞・動詞・前(後)置詞といったものは固定の記号を割り当てているが，その他に言語特有の品詞や句読点などの記号についても，記述解析に必要なものを定義している．ただし後者はあくまで解析に用いられるのみであり，単文化処理では消去される単語とみなされる．なお，日本語記述は通常分かち書きされていないため，単語としての認識に以下の基準を設けている．

・名詞列は１つの単語とする．
・動詞列はバラバラに分ける．
　例:「もっている」→「もって　いる」
・「名詞+する」は１つにする．
　例:「動作する」
・「動詞+助動詞+助動詞...」は１つにする．
　(すなわち「助動詞」としては認識しない)
　例:「補完できない」
・形容動詞は形容詞や「名詞+助詞」と捉える．
　例:「完全で」→形容詞

また，日本語では字句解析の都合上，内部で文字単位に変換される．

[用語辞書]

```
(define phrase-dict-txt
   '((単語A (単語A1 単語A2 ...))
     (単語B (単語B1 単語B2 ...))
     ...))
```

原則として「単語←→単語列（単文)」の組み合わせであるが，日本語の場合は，記述として出力される際に結合して出力される．

これら辞書情報を用いて字句・構文解析等が行われる．字句解析のアルゴリズムとしては現在のところ，日本語の場合は文字単位でマッチさせていく簡単なもの，英語の場合は単に品詞情報を付加するのみである．構文解析についてもバックトラッキングを含む単純なものであり，構文規則もそれほど複雑ではない．なお，日本語の場合は以下の構文規則で単文化を試みている．

```
<文> ::= <単文> 読点
<単文> ::= 名詞 助詞 <動詞句>
         | 副詞 <動詞句>
         | 接続詞 <動詞句>
         | 連体詞 名詞 助詞 <動詞句>
         | 形容詞 名詞 助詞 <動詞句>
         | 形容動詞 名詞 助詞 <動詞句>
<動詞句> ::= 動詞 接続助詞 <単文>
         | 動詞
         | <単文>
```

本研究ではこれまで主に日本語を扱ってきたため，前処理など言語依存部において日本語特有の処理について中心に考えてきた．現在，英語を用いる場合の各種処理部を構築中だが，原則として扱う記号等が違う以外は日本語とほぼ同じようである．また，英語(を含む多くのアルファベット圏の記述言語)は分かち書きの伝統が強いため解析システムは比較的単純で済む上，近代英語そのものが日本語よりも記号的な性質を有しており，本研究で想定している処理を行う分には特に複雑なアルゴリズムは必要ないと思われる．

6. 課題

これまで述べてきた通り，辞書情報の充実が必要である．開発を想定しているドメインの現実的なサンプル記述の収集と解析はもちろん，システムを用いながら辞書情報を蓄積することが本システムで想定されていることであり，実際の開発を意識した模擬テストが必要である．テストでは辞書情報の充実だけでなく，各ドメインごとに辞

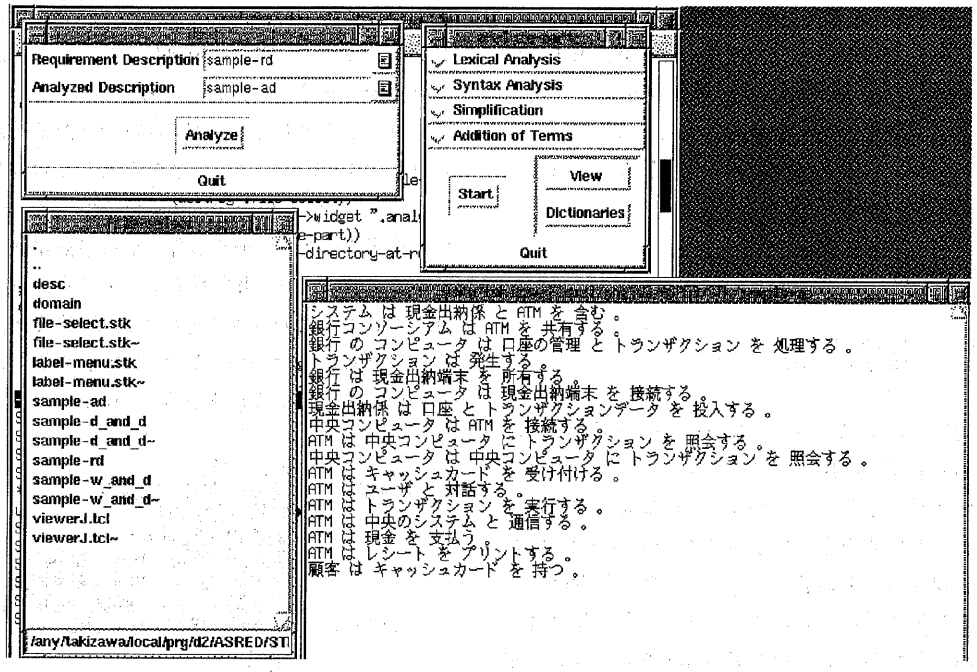

図 5 ARDES における支援ツール

書情報の検証・強化を行うことも考えられるため，さまざまな分野および被験者によるテストが必要である．

ARDES システムは現在では日本語/英語双方を記述として想定しているが，その切り分けはまだ不完全である．主な理由は構文解析部の処理が不完全だからであるが，自然言語処理技術自体が不完全である以上，この部分をより正確に処理することがそのまま最終的な結果の正確さや処理の柔軟性として反映されるとは考えにくい．むしろ，用語辞書の充実によりカバーする方が効果的かもしれないが，どの程度正確に処理を行うべきかの検証はまだ行っていない．

本研究の手法・システムは主にソフトウェア要求を想定しているが，システム全体の要求に対しても適用可能である．その場合，ソフトウェア要求の部分とそれ以外とをどのように関連付けていくかはまだ検討段階である．

参考文献

[1] 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).

[2] 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).

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

[4] 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).

[5] 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).