2R−3

# 要求仕様を対話的に獲得するシステム

野際豊朗　小谷善行
東京農工大学　工学部　電子情報工学科

## 1．はじめに

要求仕様の作成は一般に苦労の多い仕事である。要求をするのはユーザであるが、ユーザは自分の要求を説明したり文書化したりする能力がないのが普通である。したがって開発者がユーザと共同して行うことが望ましい。しかし現実には、この苦労を避けた結果、両者の間に十分なコミュニケーションが取れず、真の要求をくみ取れない状況がしばしば発生する。

本稿では、ユーザが対話的に要求仕様を作成するための知識処理システムを提案する。これは開発者とユーザの間の話し合いをシミュレートするものである。開発者の部分をこのシステムが受け持つ。システムは、ユーザとの機械対話インタビューにより、情報収集を行い、要求仕様を内部に生成する。

## 2．設計方針

本システムの対象は、新たに開発するコンピュータソフトウェアシステム（以下開発ソフトウェア）を使用するユーザである。ユーザは、システムとの対話を通じて、解くべき問題を分析していく。その結果として得られる、開発ソフトウェアの処理概要、目的、用途、処理する対象の関連知識などの知識を収集し、所定の形式で知識ベースを作成する。

ユーザの要求を単にまとめただけでは、情報があいまいさで不足しているため、良い要求仕様にはならないことが指摘されている。[1] [2] その点から、結果に対しては無矛盾性、完全性等を要求する。

処理そのものはなるべく単純にし、知識獲得の動作、推論方法やその結果等は知識の構造・内容に帰着させることで知識の体系化し、システムに柔軟性を与える。

## 3．システムの概要

本システムは、図1に示すように、4つの知識ベースとそれらを元に推論、学習する処理からなる。各知識ベースの内容は以下のとおりである。

① ユーザ要求ベース：ユーザとの対話によって得た要求を開発ソフトウェアの目的・用途等の項目について所定の形式でまとめたもの
② 関連知識ベース：　開発ソフトウェアの適用分野についての知識を含んだ一般知識
③ 処理情報ベース：　プログラミングについての知識や、要求仕様としてふさわしい表現を集めたクラス知識
④ 要求仕様ベース：　結果となる要求仕様

①②はユーザ側の知識であり、ある程度のあいまいさを認め、そのかわりに広い範囲の推論を行わせる。これによって、知識入力の省力化を図ると共に、一時的に矛盾した知識を提供することによってユーザの誤解発見を促す役割を果たす。

処理要求ベースは、ユーザに対してプログラミングの知識などを提供し、また、結果となる要求仕様のクラスフレームとなる。そのため、これは形式的で厳密な知識だけを完全に分類された形で持つ。このような性格から、ユーザは参照するだけである。

処理情報ベースは、さらに、基本概念、要求項目、処理知識、制御知識の4つに分かれる。

基本概念は要求仕様に記述可能なオブジェクトの概念を提供する。「要求項目」は、ユーザが入力すべき要求項目およびその入手方法についての知識である。処理知識は本システムがあつかうことのできる処理をデータフローグラフの形で記述した知識である。

処理知識には「目的」の項目があり、複数存在する同様の処理方法の中から最善の物を選ぶために用いられる。その評価項目として、「効果」が「目的」に応じて分類されながら記述される。

制御知識には、処理知識に対する制限や効果の定量化方法などの推論を制御するためのメタ知識が置かれる。

## 4．知識としてのデータフローグラフ

ユーザが必要とする処理を示すためや、要求する機能をどのように実現するかといった知識情報としての「処理」を扱う場合、その表現としてデータフローグラフを使用する。内部的には図2のようなフレームで表現される。データフローグラフは、コンピュータ処理で必要な、入出力、データベース、処理の3要素が記述できるので、どのような処理で要求する機能が実現できるかというプログラミングの知識を提供するには便利である。

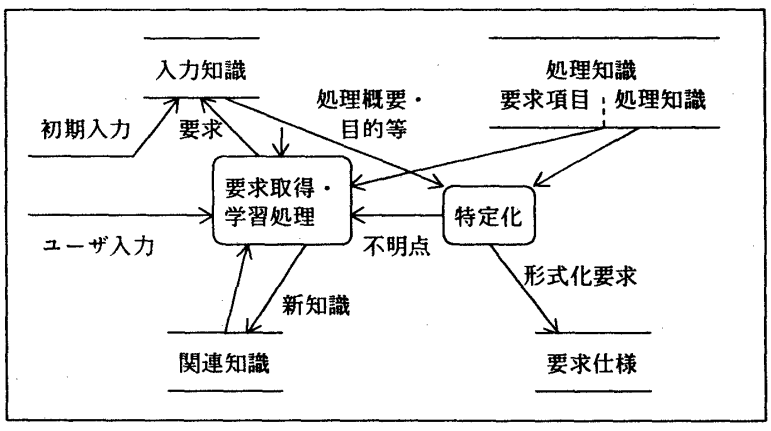


図1　本システムの概要

An Interactive System of Acquiring Requirements Specification
Toyoro Nogiwa, Yoshiyuki Kotani
Tyokyo university of Agriculture and Tecnology

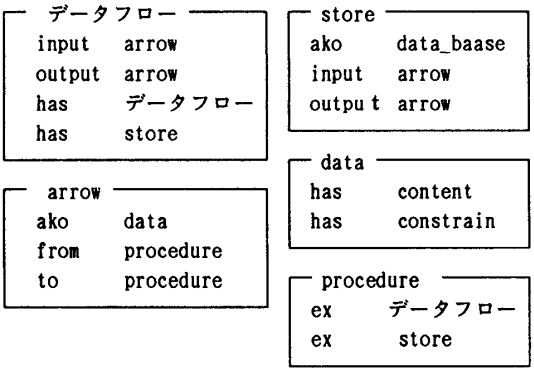


図2　データフロー図のフレーム構造

本システムでのデータフローグラフはストアや処理を結ぶ矢印がデータの制約等を持つ。

また、データフロー図は単なる知識だけではなく、直観的に理解でき、特別な知識を必要としない点から、処理内容をユーザに説明する視覚的な情報としても扱われ、本システムが生成した処理のレビューとなる。

## 5．処理概要

本システムは、ユーザの要求に基づいて処理情報ベースから最適な処理の組み合わせを選び出し、それから導き出される効果、制約等が実際の目的に適合するかどうかをユーザに問うものである。

本システムの処理は大きく分けて2つある。一つは情報の収集である。必須処理の概要、目的、現行の伝票等、本システムを使用前に確定している情報は事前に入力しておくことが望ましいが、そうできない場合や与えられただけでは情報量が不足している場合には処理情報ベース中の「要求項目」に基づいてシステムが処理対象など必要とする情報を求めてくる。また、ユーザ入力中に未知の概念が存在する場合、その概念が処理情報ベース中の概念で説明されるまでユーザに対して問いかける。

例として、酒類販売会社の倉庫の問題[4]において、酒の知識を得る様子を説明する。

「倉庫」は一般的な概念として処理情報ベースに存在する。ここで、倉庫の知識から、それが必要とするものが、保管の対象であることを知り、それ（ビン詰めの酒）をユーザに尋ねる。さらに、管理の最小構成（酒そのもの）、単位（個数）、識別方法（銘柄）を推論あるいは質問によって得る。これらも倉庫が要求する知識である。結果として、酒の実体がが何であっても処理をするための知識は得られる。

このように、集めた情報が内部の処理を特定できるようになると、その処理を制約等の条件と共にユーザに提示し、目的適合するかどうかをユーザに問う。あるいは、細部が不明確な場合にはそれを明確にするために要求する品質や実現できる制約などをユーザに尋ね、処理やその細部を決定していく。その結果特定化された処理情報フレームは要求仕様として格納される。これがもう一方の処理である。

推論によって単一のフレームに相反する情報を持ったスロットが現れた場合、あるいは、単一のスロットしか認められないフレームに対して複数の同一スロット名が現れた場合、これを矛盾として、その推論課程を表示し訂正を促す。しかし、一般に「相反しない」ことの証明を得ることは難しい。これについては知識構造で解決する。つまり、フレームの構成要素を慎重に定義し、前もって場合分けして置くなどで、本質的に矛盾の起こらない、あるいは検査の容易なフレームやスロットを作るということである。処理情報ベースはユーザによる改変を受けないので、知識構造が破壊される恐れはない。

## 6．評価

本システムの特徴は要求仕様の記述者をユーザにおいた点である。そしてユーザの作成した要求仕様を妥当なものとするために、厳格な知識ベースである処理情報ベースを結果のクラスフレームとした。ここに記述される処理知識が実現可能なものであれば作成された要求仕様も実現可能である。

処理知識と共にその制約、効果等を記述し、ユーザの要求によって生成された処理の品質、制約などが導出できる。また、知識の不足から仕様の不完全性を識別することができる。さらに、処理をデータフロー図として表示することで、作成された仕様が目的にあったものであるかどうかをユーザが判断することができる。

ユーザは、入力要求を変更することも考えられる。この場合、推論を再度行う手間はかかるが、質問に対する返答の知識は流用できるので、仕様変更に対する負担は大幅に軽減する。本システムは、この過程を繰り返すことで真の要求を引き出すことを狙っている。

このように、用意した処理から適したものを選び出し組み合わせることは大きな利点を与えるが、一方で、用意されない処理は行えない欠点も持つ。これは、設計者が新たに処理知識を与えることで解決する。この際、ユーザによって入力された要求等の知識を利用することができる。この操作はユーザと設計者の対話と位置づけることができる。動作の大きな部分を知識ベースの構造や内容によっている本システムはこのように大きな拡張性を持っている。

出力は本システムによって作成されたフレーム表現の知識情報そのものである。これは、一種の形式的仕様と見なすことができる。また、知的システム設計ツールを想定した場合、その入力として最適である。

## 7．まとめ

ユーザが要求仕様を作成するシステムについて提案を行った。目的に応じた各種の知識ベースを用意し、柔軟で豊富な知識を用いた推論から、仕様としての厳密な知識を得ることができる。視覚的な理解を含ませるために、知識表現のひとつとしてデータフローグラフを採用した。動作を知識構造に帰着させることで拡張性に富んでいる。

本研究には、科学研究費、重点領域「高信頼性高品質ソフトウェアの構成原理の研究」の援助を受けている。

参考文献

[1] 花田收悦: ソフトウェアの仕様化と設計, 日科技連, 1986.

[2] 有澤誠: ソフトウェアプロトタイピング, 近代科学社, 1986.

[3] Capt. Hilliard Holbrook III: A Scenario Based Methodology for Conducting Requirement Elicitation, Software Engeneering Notes, 15-1, pp.95-104, 1990.

[4] 山崎利治: 共通問題によるプログラム設計技法解説, 情報処理, 25-9, pp.934, 1984