〔公　開〕

ＴＲ－Ｃ－０１００

設計知識の構造化と活用
― 設計知識の再利用 ―

安達　久人
Hisato　ADACH

１９９４　　８．３０

ＡＴＲ通信システム研究所

設計知識の構造化と活用
― 設計知識の再利用 ―

平成６年８月

安達　久人
ＡＴＲ通信システム研究所
〒619-02　京都府相楽郡精華町光台2-2
Tel : 07749-5-1249
Fax : 07749-5-1208
e-mail : hisato@atr-sw.atr.co.jp

目次

1. はじめに
2. 設計プロセスモデル
3. 設計支援方式
   3.1 設計手順提示機能
   3.2 設計内容提示機能
4. 評価
   4.1 生産性の評価
   4.2 再利用性の評価
   4.3 分析
5. まとめ
6. 参考文献

図，表

付録1 ドキュメント保守支援方式への適用

付録2 視点利用規則

付録3 知識工学分野における本研究の位置付け

1. はじめに

近年，通信事業者による画一的なサービス提供に代わって，サービス利用者の要求に合わせたサービス提供が求められている．このため，サービスを実現する通信ソフトウェアもサービスの提供形態の変化への対応が求められるとともに，今後ますます生産性の向上が求められることが予想される．

ソフトウェア設計では，生産性の向上を目的とした様々な設計法が提案されており，通常，設計者はこれらの設計法に従って設計を行っている．設計法には設計法毎に決められた最終生産物を作成するために，どのような設計項目をどのような順番で設計すれば良いかが大まかに示されている．例えば，構造化分析手法[1]では，最終生産物はデータフローダイヤグラムであり，そのためにコンテキストダイヤグラムの作成，データ源泉，データ吸収の分析等を行うよう指示されている．設計法に示された設計項目について設計するためには，設計法には示されていない細部の設計項目について設計した結果を参照する必要がある．

このような細部の設計項目はソフトウェアの開発依頼者の要求やソフトウェアが動作するハードウェア環境など様々な制約を考慮して行う必要があり，設計者にはこれらの知識やノウハウを持っていることが要求される．

設計に必要な知識やノウハウの不足は，認識すべき制約の洩れ（つまり，設計の洩れ）による設計の手戻り等を引き起こしソフトウェアの生産性を阻害する原因の１つになっていると考えられる．従って，このような知識やノウハウの活用法の確立がソフトウェアの生産性の向上には急務であると考えられる．

このような問題に対して，設計作業を行う上で必要な知識を予め用意することを目的とした研究が行われている．例えば，プロセスプログラミング[2]の研究は，設計作業を分析し設計上の知識を獲得することにより，予め設計手順を記述しようとするものである．しかし，設計の多様性の問題から，手順の詳細な部分を一般的な知識として記述することは困難であり，これらの研究では設計法レベルの手順の記述が限界であると考えられる．

このようなアプローチに対し，ソフトウェアの設計時に考慮された様々な情報を記録し，他のソフトウェア設計で再利用する考え方が注目されている．

我々の研究は，この考え方に基づいたものであり，設計過程（制約の認識と制約を満足する解の検討結果）を記録し，これを再利用することにより，新規設計で設計者に「考慮すべき制約とその制約と依存関係を持つ他の制約」を示すとともに，制約を満足する解の候補として「過去の事例では設計内容がどうなっているか」を示すものである．

このような支援を行うためには，これまで残されていなかった設計者の設計過程に関する情報を残す必要があり，そのために記録する情報とその記録形式を明確にする必要がある．

記録情報に関して，設計結果に対する根拠の情報として賛成，反対の議論を設計時に記録するWEB [3]，IBIS (Issue Based Information System) [4]をベースとした記録方法 [5][6]などが提案されている．しかし，これらの情報から我々が目的とする支援を行うためには，記録を人間が読んで，参照した設計項目を洗い出してから手順を考える必要があり，また，類似事例を検索するためには全ての事例について同様の作業を行う必要がある．従って，これらの研究で提案されている記録情報と記録形式を，我々が目的とする支援を行うために用いるのは困難であると考えられる．

また，設計手順の再利用に当たって，設計手順を時系列に従って記録し利用する方法が考えられるが，この方法では設計時の参照漏れによる設計の手戻りの問題に対処することは困難である．

我々は，設計手順を記録された順番ではなく，参照関係から抽出することで，設計時の参照漏れによる設計の手戻りの影響を受けることなく，設計の手順と参照すべき情報を提

示できると考えている．

この考えに従って設計者の設計過程を，設計者が捉えたあるまとまり(設計者がどのような設計項目について設計し，その結果はどうなったか)とまとまりの間の参照関係としてモデル化した設計プロセスモデルを考案した[7]．

本稿では，まず，設計プロセスモデル(2章)を示す．次に記録した情報の蓄積利用方法とそれに基づく設計支援システムの概要を示し(3章)，本方式の評価を行った結果を示す(4章)．

2. 設計プロセスモデル

一般に，ソフトウェアの設計で一度に人間が認知できる機能やモジュールの大きさには限界がある．このため，設計を行う場合，通常，一時に一部分の機能やデータのみが詳細化される．

設計者が詳細化を行う場合，まず，詳細化の対象(機能，データ等)を限定し，詳細化の観点(機能構成，入／出力データ等)を決定する．次に詳細化する内容を原要求(ソフトウェアの開発依頼者が提示した要求内容)やそれまでの詳細化結果(設計内容)を参照して検討する．

我々はこのような設計作業を，詳細化の対象，詳細化の観点および設計内容の三つ組で表わされるまとまり（設計者の思考単位）とその設計において参照した原要求や他の思考単位との参照関係として捉えモデル化を行った(設計プロセスモデル(図1)と呼ぶ)．

本モデルでは，詳細化の対象をキー設計エンティティ，詳細化の観点を視点と呼ぶ．キー設計エンティティと視点のペアを設計指標と呼ぶ．設計指標と設計内容で表わされる思考単位を設計ビューと呼ぶ．

ある設計ビューを設計する際に原要求または他の設計ビューを参照して設計したという関係を利用関係と呼ぶ．なお，原要求はソフトウェアの設計依頼者から与えられるものであり，詳細化した結果ではないが，利用関係を統一的に管理するため，設計ビューと同一の構成とした．この場合，キー設計エンティティは「在庫管理ソフトウェア」といったようにソフトウェアの名称，種類を表わし，視点は「原要求」となる．詳細化の内容には要求内容が記述される．

設計内容を構成する要素は設計エンティティと呼ぶ．設計エンティティの幾つかは更に次の詳細化の対象(キー設計エンティティ)となる．例えば，出庫処理を機能項目という観点から詳細化した結果の構成要素(出庫依頼の入力，出庫指示書の作成，…)は設計エンティティとなる(図1)．このうち，「出庫指示書の作成」について，更に，ある観点から詳細化を行う場合，「出庫指示書の作成」はキー設計エンティティとなる．

3. 設計支援方式

通常，設計者が設計を行う場合，設計すべき設計項目とその際参照すべき他の設計項目の設計結果を洗い出し，それらをもとに設計手順を決定した後，個々の設計内容を検討する．

我々が提案する設計支援方式では，このような設計で必要とされる活動に合わせ、設計手順(設計手順提示機能)と設計結果の提示(設計内容提示機能)を行う．

本方式では，提示する設計手順および設計内容は過去に設計された事例のうち，最も今回の設計に類似したものを提示することを目的としており，このため，本方式では，設計者の思考過程を記録した設計プロセスを複数事例について集め構造化した設計事例データベースを用いる．

我々は，本方式を実現するプロトタイプシステムの試作を行った．図2に本システムの構成を示す．以下，本システムを構成する設計事例データベース，設計手順提示機能，設計

手順提示機能で用いる視点利用規則，設計内容提示機能についてそれぞれ記述する．

設計事例データベース

設計事例データベースは，設計プロセスモデルに基づいて記録された過去の設計事例を問題領域毎に集め整理したもので，過去に設計された事例の中で今回の設計に最も類似していると考えられる設計手順や設計内容の提示に利用する．設計手順や設計内容は設計指標を検索のキーとして用いることで実現する．このため，設計事例データベースは，同一設計指標を持つ複数の設計結果(バリエーションと呼ぶ)をグループ化し，バリエーション間の利用関係と共に蓄積することとした．在庫管理プログラムの設計事例2つを格納した設計事例ベースの例を図3に示す。例えば、<出庫依頼 , 媒体> が、電話や出庫依頼書(紙)による場合(図中a)と、e-mailによる場合(b)とで、<出庫処理 , 機能項目> に対して、"出庫依頼の入力"(c) または、"出庫依頼を構文解析する"(d) の各々を設計したという事例を表している。

3.1 設計手順提示機能

本機能は，設計の各段階で設計すべき設計指標を提示する．設計指標には設計法で規定される大まかな手順に相当するもの(設計目標と呼ぶ)と設計目標間の詳細手順に相当するものとがある．設計手順提示機能は，それぞれに対応した設計目標提示機能と設計指標提示機能の2つから構成される．以下，それぞれの機能について述べる．

設計目標提示機能

本機能では，設計法で示されている設計生産物を設計目標として提示する．

設計法で示されている設計生産物の間には依存関係があり，ある設計生産物を設計できるためには，他の設計生産物の設計が完了していなければならないという性質がある．この関係を視点利用規則として予め記述し，現在の設計で，どの設計生産物までが設計されているのかを，その規則により調べ，次に設計すべき設計生産物を設計目標として提示する．

以下，本機能で用いる視点利用規則と設計目標の提示方法を示す．

視点利用規則

視点利用規則は設計法ごとに作成する．規則は，設計すべき設計生産物(中間生産物を含む)と設計目標が設計できるための条件から成り，

設計生産物X :- 設計生産物1 , ... , 設計生産物n.

(右辺(設計生産物1～n)が設計されていれば，左辺(設計生産物X)が設計できる)と表す．

視点利用規則における設計生産物は，前述の設計プロセスモデルに合わせ，キーと視点の対で記述する．

本規則における視点には設計法で示されている設計生産物名が該当し，キー設計エンティティについては変数による記述を可能とする．例えば，「ある機能について，入力データと出力データという視点からの設計が完了していれば，その機能のプロセス構成が設計できる」という依存関係は次のように記述する．

<*key , プロセス構成> :-

<*key , 入力データ> , <*key , 出力データ>.

*key は変数

設計目標の提示方法

視点利用規則に基づく，設計目標の提示方法を以下に示す(図4).

システムは視点利用規則の集合から，「現設計プロセスにおいて左辺が設計されていて，右辺が設計されていない」という条件を満たす規則を限定し，その右辺に記述されている設計生産物を設計目標として提示する．

例えば，前述の視点利用規則が上記条件を満たすかどうかは以下のようにして調べる．

右辺の評価

- 現設計プロセスから，視点"入力データ"を持つ設計ビュー全てを検索する(図4の例では(a)が見つかる)．
- 見つかった設計ビューのキー設計エンティティ(例では"出庫処理")を*keyの候補とする．
- 現設計プロセスから，<出庫処理,出力データ>を持つ設計ビューを検索する(図4b)が見つかる)

左辺の評価

- キー設計エンティティ"出庫処理"，視点"プロセス構成"を持つ設計ビューが，現設計プロセスに存在しなければ設計目標とする(図4(c)が設計目標となる)．

設計指標提示機能

本機能は，設計の多様性の問題から，予め規則として記述することが困難な具体的な手順を，設計目標の間を埋める具体的な手順として提示する．この具体的な手順は，設計事例データベースに記録されている設計指標間の利用関係から求める．

以下，設計指標の提示方法を示す(図5)．

設計指標の提示方法

① システムは，設計事例データベース中から設計者が選択した設計目標と一致するものを探す(図5(b)が見つかる)

② システムは，一致した設計目標との利用関係を順にたどり，設計手順を表す木(設計展開木と呼ぶ．図5ワークエリア中の木構造)を作る．システムは，設計展開木の生成を抑制するために，利用関係をたどる際に現設計プロセスを調べ設計済の箇所(図中網掛け部分)を検出した場合は，それ以降の利用関係をたどる作業を中止する

③ システムは，設計展開木をもとに次に作成すべき設計指標およびその設計で参照すべき設計指標(参照指標と呼ぶ)を提示する(図5の場合，設計手順は以下の順序で提示する)

④ <出庫処理,機能項目>を<出庫処理,入力データ>と<出庫処理,出力データ>を参照して作成する

⑤ <出庫処理,プロセス構成>を<出庫処理,機能項目>と<在庫管理,原要求>を参照して作成する

3.2 設計内容提示機能

設計指標提示機能で提示された設計指標に対応する設計結果のバリエーションが設計事例データベース中に複数存在した場合，現在の設計に最も類似し，再利用の可能性が高いバリエーションを検索し提示する．

設計結果の検索は設計指標を検索キーとして行うが，検索結果は一般に複数であるため，検索結果を絞り込むための評価基準が必要となる．

我々は，この評価基準として，事例の一致総数と事例総数の2つの基準を設けている．

以下，設計結果の提示方法を示し，次に各絞り込みの基準に基づく絞り込み方法につい

て示す.

設計結果の提示方法

(1) システムは設計事例データベースから，設計者が選択した設計指標と同じ指標の設計結果のバリエーションを全て求める(図6網掛け部分)

(2) システムは，各バリエーションの利用関係を調べ，設計内容提示機能で提示された参照指標と同じ指標を持つバリエーションを限定する(図6(a)，(b))

(3) システムは限定されたバリエーションを絞り込みの基準(後述)に従って絞り込み，そのバリエーションを参照して設計した設計指標を持つバリエーションを設計者に提示する(図6では(b)が提示されたことを表わしている)

(4) 設計者は，提示されたバリエーションを参照して設計指標の設計を行う

設計事例の一致総数による絞り込み

「現在の設計指標を詳細化する際に参照すべき設計ビューと過去の設計で参照した設計ビューの一致度が高ければ，後者の設計ビューに基づき設計された結果の再利用が期待できる」という考え方に基づいて絞り込みを行う．本基準では「設計結果の類似性」を「設計エンティティの一致度」として捉える．設計事例の一致総数による絞り込みの方法を以下に示す(図7).

(1) 絞り込みの対象を限定する

・設計結果の提示方法の(1)～(4) を参照

(2) 各参照指標の設計結果の類似性が最も高いものを調べる

・一致総数はバリエーション毎に求める

・酒問屋の事例は<受付係システム,入力データ>の一致数が2で<受付係システム,出力データ>の一致数が2．従って一致総数は4となる

・飲食店の事例は<受付係システム,入力データ>の一致数が1で，<受付係システム,出力データ>の一致数が1．従って一致総数は2となる

・一致総数が最も多い(類似性が高い)のは酒問屋の事例

設計事例の事例総数による絞り込み

設計事例の一致総数が同数の場合，「複数のバリエーションのうち，過去に何度も設計されたものは今回の設計でも利用される可能性が高い」という考えに基づき絞り込みを行う.

設計事例の事例総数による絞り込みの方法を以下に示す(図8).

(1) 事例の一致総数が同じバリエーションについて，過去に設計された回数を調べ，設計された回数が最も多いバリエーションを求める

・酒問屋の事例は事例総数が3

・酒屋の事例は事例総数が2

・事例総数が最も多いのは酒問屋の事例

4. 評価

筆者らは，ソフトウェアの設計実験を行い，提案する手法のソフトウェア設計における効果を評価した．実験は要求分析工程について行い，以下のデータを測定した.

(1) 生産性

設計ガイド機能の使用による工数削減効果の評価を目的とし，DIGによるガイド有りと

ガイド無しの場合の設計工数（総設計時間）を測定し比較する．

(2) 再利用性

設計事例の再利用性の評価を目的とし，設計手順（設計目標および設計指標）の再利用数と設計内容の再利用数をそれぞれ測定する．

以下，それぞれの評価結果について示す．

4.1 生産性の評価

ガイド有DIGを用いることにより，総設計時間で4時間の時間削減．時間削減率は4時間／24時間（約17%）であった(表1)．この結果より，ガイド有りDIGの方が問題が複雑であるにも拘わらず設計時間が短い（設計結果の詳細化の度合は両者で同程度）ということが言える．

4.2 再利用性の評価

事例の再利用性の評価は設計手順の再利用性と設計内容の再利用性の2つに大きく分けられる．設計手順は大まかな手順を示す設計目標と設計目標間の間を埋める詳細な手順を示す設計指標の2つに大きく分けられる．このうち，設計目標は視点利用規則を用いたものであり100%利用されるため，評価対象から外し，設計指標の再利用数のみを測定した．

設計指標の再利用性

表2に設計指標の再利用性に関する実験結果を示す．実験結果より，1つの設計目標に対して提示される詳細な手順のうち約4割が再利用されていることがわかる．

設計内容の再利用性

表3に設計内容の再利用性に関する実験結果を示す．

実験結果より，1つの設計指標の設計に対して約1割強の設計エンティティが再利用されていることがわかる．

4.3 分析

生産性について，ガイド有りの設計がガイドなしの設計よりも総設計時間が短縮されたのは，以下の理由によると考えられる．

(1) 参照物の検索時間の減少

(2) 再利用による編集時間の減少

(3) 手順または内容について新たに検討する時間の減少

(1) は過去の事例を参考に設計を行う際，膨大な過去の事例の中から類似する事例を見付け出す必要があるが，我々の提案する手法では事例を問題領域毎に蓄積していることと，設計指標を検索のためのキーとして用いていることで検索の効率がガイド無しDIGの設計よりも高いためであると考えられる．

(2) は，一般に設計時に情報を記録する場合記録の工数がかかるが，設計内容そのものを再利用することで記録の工数が減少したためであると考えられる．

(3) は，設計者は設計すべき設計指標とその設計で考慮すべき設計指標を洗い出し設計手順を決定する必要があり，実際の設計ではこの検討にかかる時間が多いと考えられる．ガイド有りのDIGを用いることによりこの検討に要する時間を減らすことができたためであると考えられる．

また，再利用性の評価の際に再利用の効果以外に，『事例を見ることで設計の洩れ，原要求の洩れに気付く』，『設計エンティティの命名法について事例が参考になった』という報告が被験者よりなされた．特に前者は設計の洩れによる手戻りの防止に効果があるこ

とを示しており生産性向上への効果も期待できると考えられる．

5. まとめ

本稿では，設計時に設計者の設計過程を記録し再利用することによる設計支援方式の提案を行った．

具体的には，設計支援で用いる設計過程の記録形式として設計プロセスモデルを提案し，本モデルに従って記録した設計事例を新規設計支援で利用する方法について提案した．

また，本方式に対する評価実験を行った結果，生産性で２割弱の時間短縮の効果が見られるとともに再利用性については，設計方針で約４割，設計エンティティの再利用性については約１割の再利用効果が見られた．

設計エンティティの再利用性が若干少ないことを除き，本手法がソフトウェアの設計における生産性の向上にある程度効果があることが実験を行うことで明らかにすることができたと考えられる．

今後の課題として，本方式では設計手順の提示に関して設計すべき設計指標とその設計で参照する設計指標の提示のみを行っているが，評価実験を通して設計手順，および使われる用語の理解性に問題があることがわかった．設計手順の理解性については，大まかな手順（設計目標）に関する理解性と詳細な手順（設計指標）の理解性の問題がある．前者は，今回の提示方法では一覧性に欠けることが問題であり，これは，前後の設計目標を提示し，設計の流れを把握させる，または，一覧の提示機能を設けることで解消できると考えられる．後者は，『何故設計しなければならないか？』という理由に関する理解性の問題を含んでおり，何らかの説明のための方法の検討が必要であると考えられる．

用語の理解性については，事例ベース構築時に事例ベース管理者による説明の付加，用語の説明用辞書の整備等の必要がある．

6. 参考文献

[1] Tom DeMarco, Structured Analysis and System Specification, YOURDON, inc, 1979

[2] L.Osterwell, Software Processes are Software Too, Proceedings of the Ninth International Conference on Software Engineering, 1987

[3] D.Knuth, Literate Programing, The Computer Journal, Vol.27, No.2, 1984

[4] J.Conklin, M.Begeman, gIBIS : A Hypertext Tool for Exploratory Policy Discussion, ACM Transactions on Office Information Systems, Vol.6, No4, October 1988

[5] C. Potts and G. Bruns, Recording the Reasons for Design Decisions, Proceedings of 10th ICSE, 1988

[6] J. Lee, Extending the Potts and Bruns Model for Recording Design Rationale, Proceedings of the 13th International Conference on Software Engineering, 1991

[7] Masaki HAMADA,Hisato ADACHI,Recording Software Design Process for Maintaining the Software,Proceedings of COMPSAC93,1993

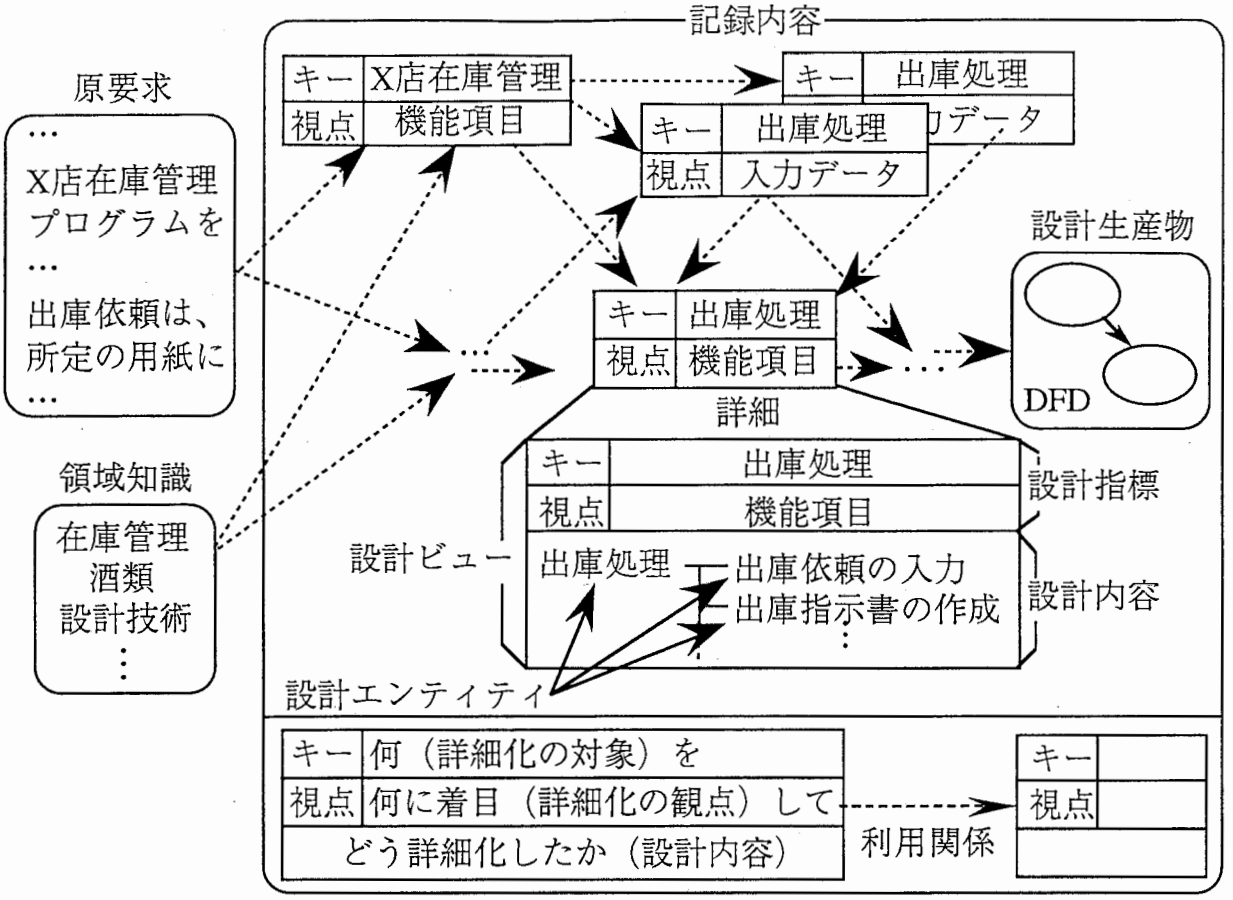

図１　設計プロセスモデル

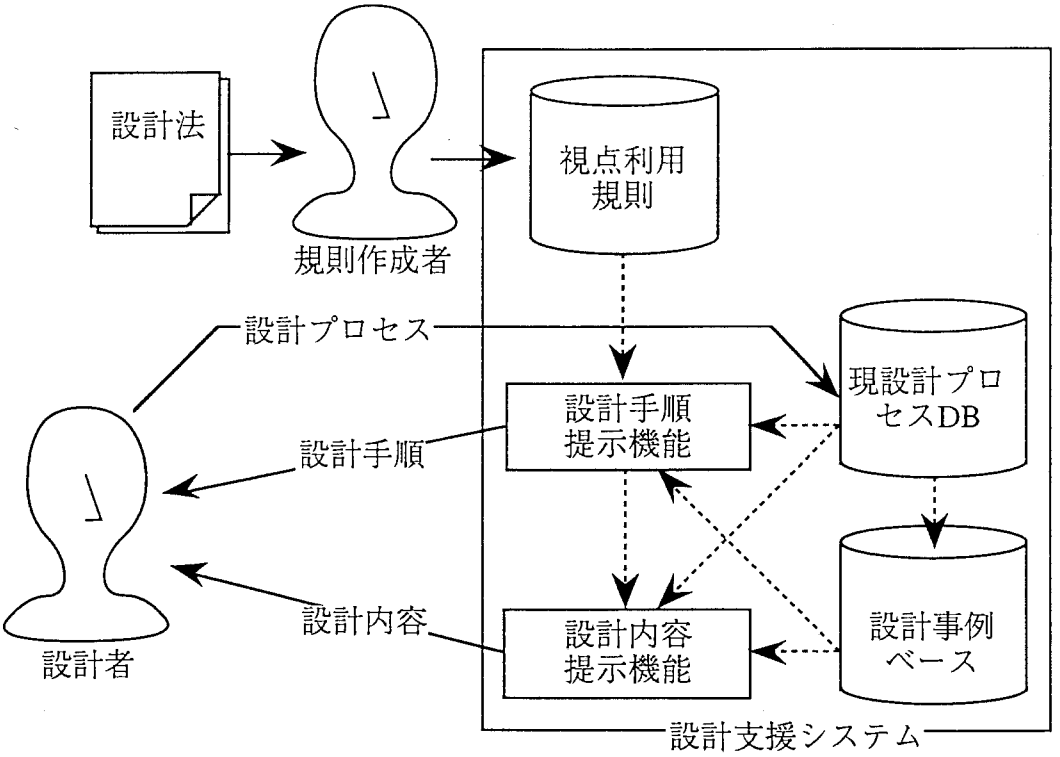
設計法
規則作成者
視点利用
規則
設計プロセス
現設計プロ
セスDB
設計手順
提示機能
設計手順
設計者
設計内容
設計内容
提示機能
設計事例
ベース
設計支援システム

図2　設計支援システム

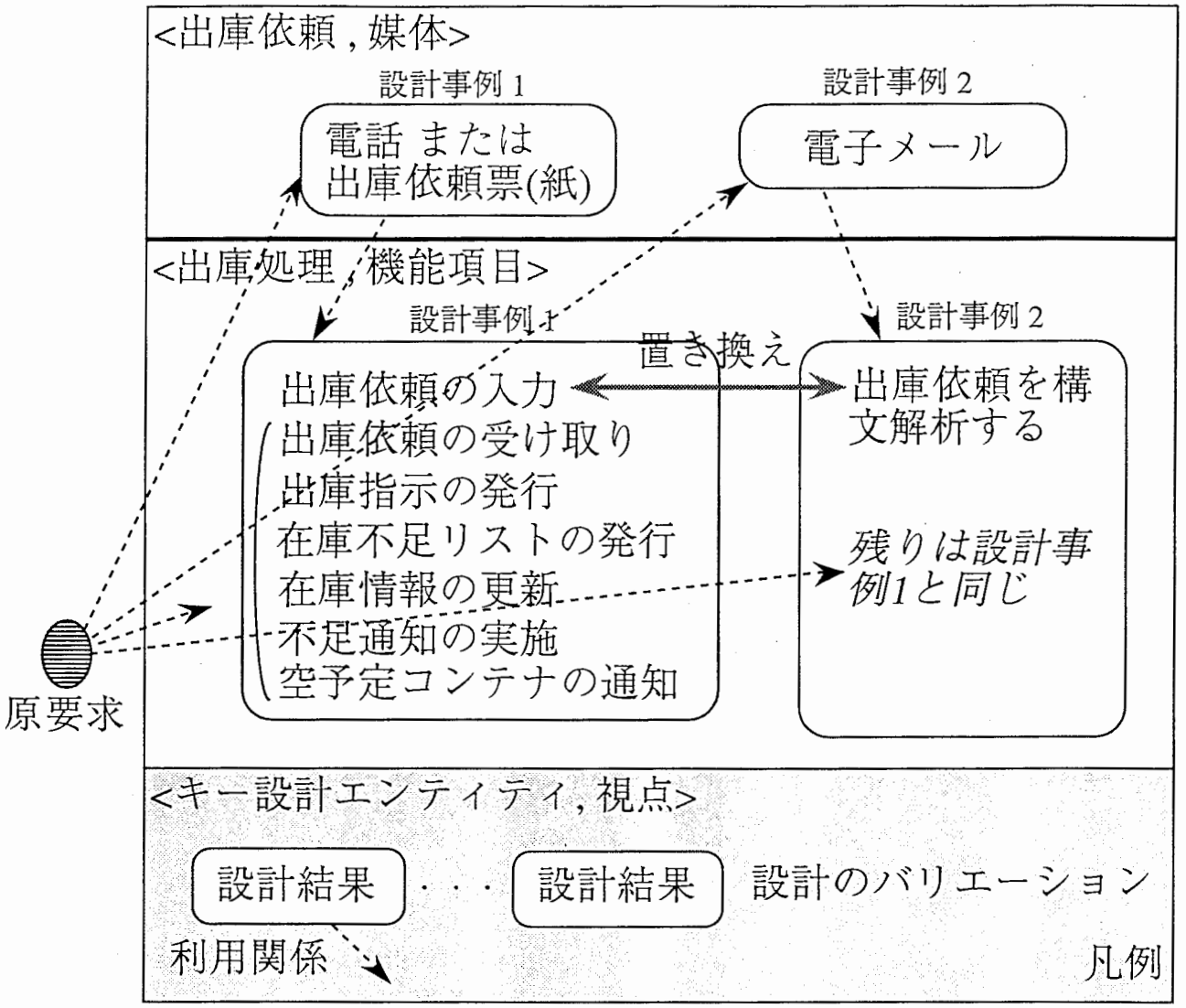
<出庫依頼，媒体>
設計事例 1
電話 または
出庫依頼票(紙)
設計事例 2
電子メール
<出庫処理，機能項目>
設計事例 1
設計事例 2
置き換え
出庫依頼の入力
出庫依頼の受け取り
出庫指示の発行
在庫不足リストの発行
在庫情報の更新
不足通知の実施
空予定コンテナの通知
出庫依頼を構文解析する
残りは設計事例1と同じ
原要求
<キー設計エンティティ，視点>
設計結果
・・・
設計結果
設計のバリエーション
利用関係
凡例

図３　設計事例ベース

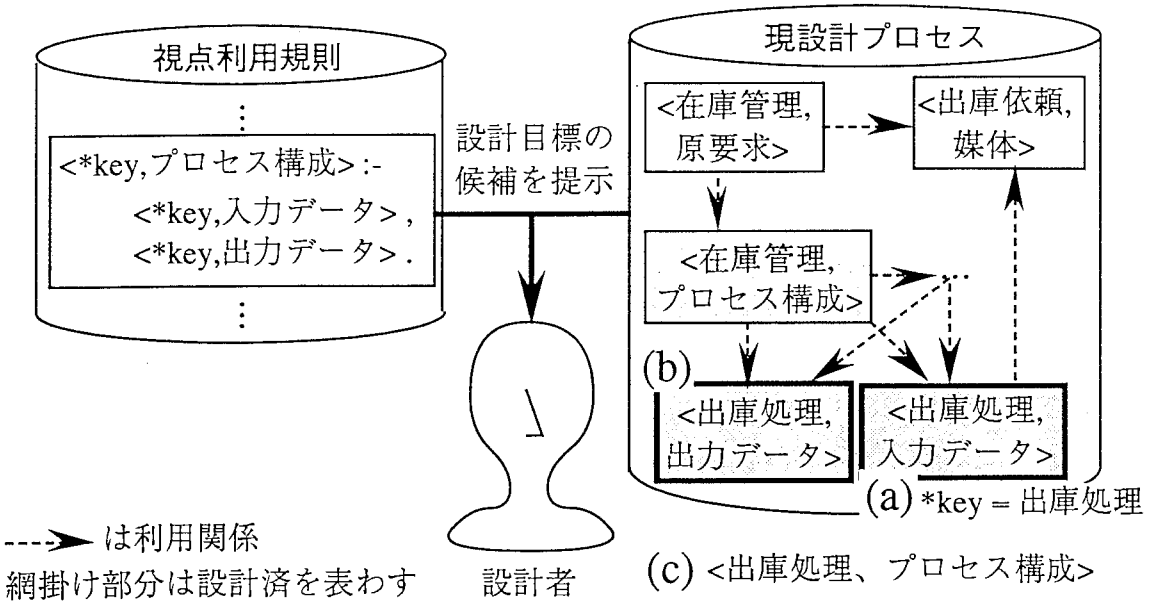

図 4　設計目標の提示方法

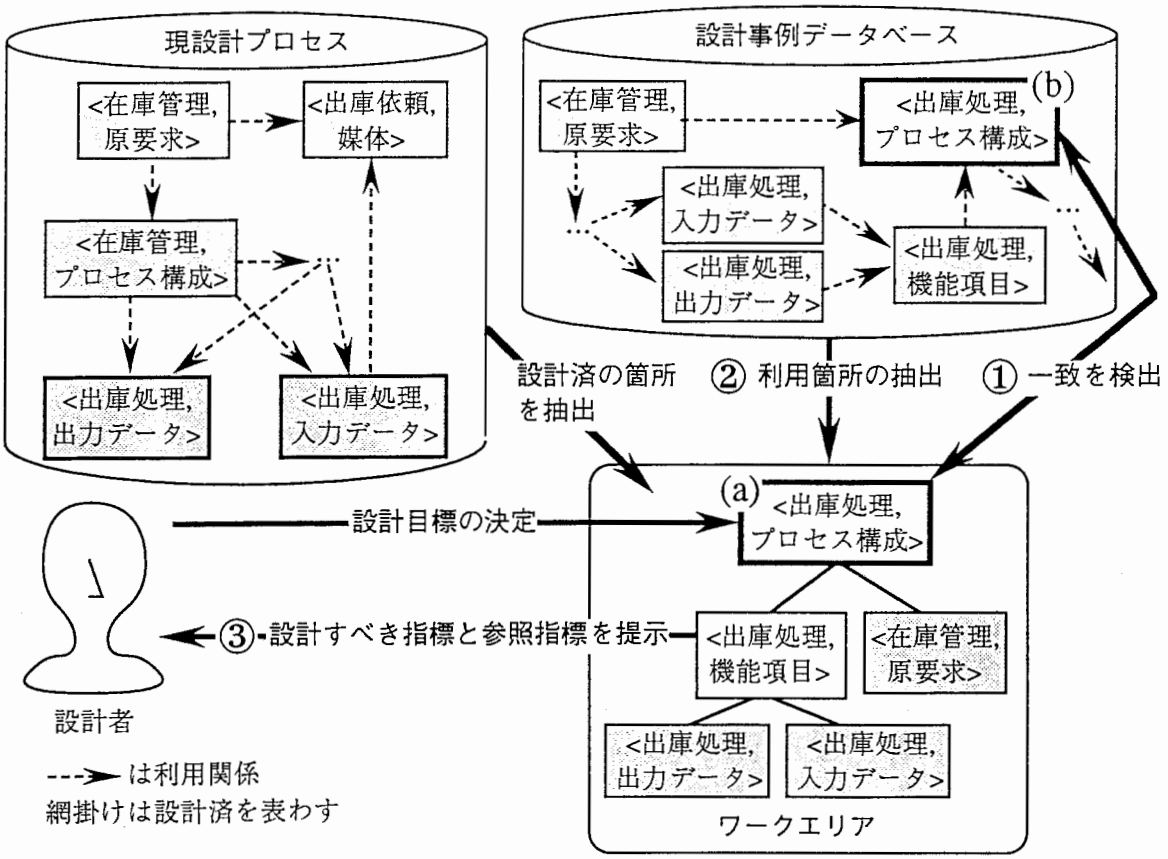

図５　設計指標の提示方法

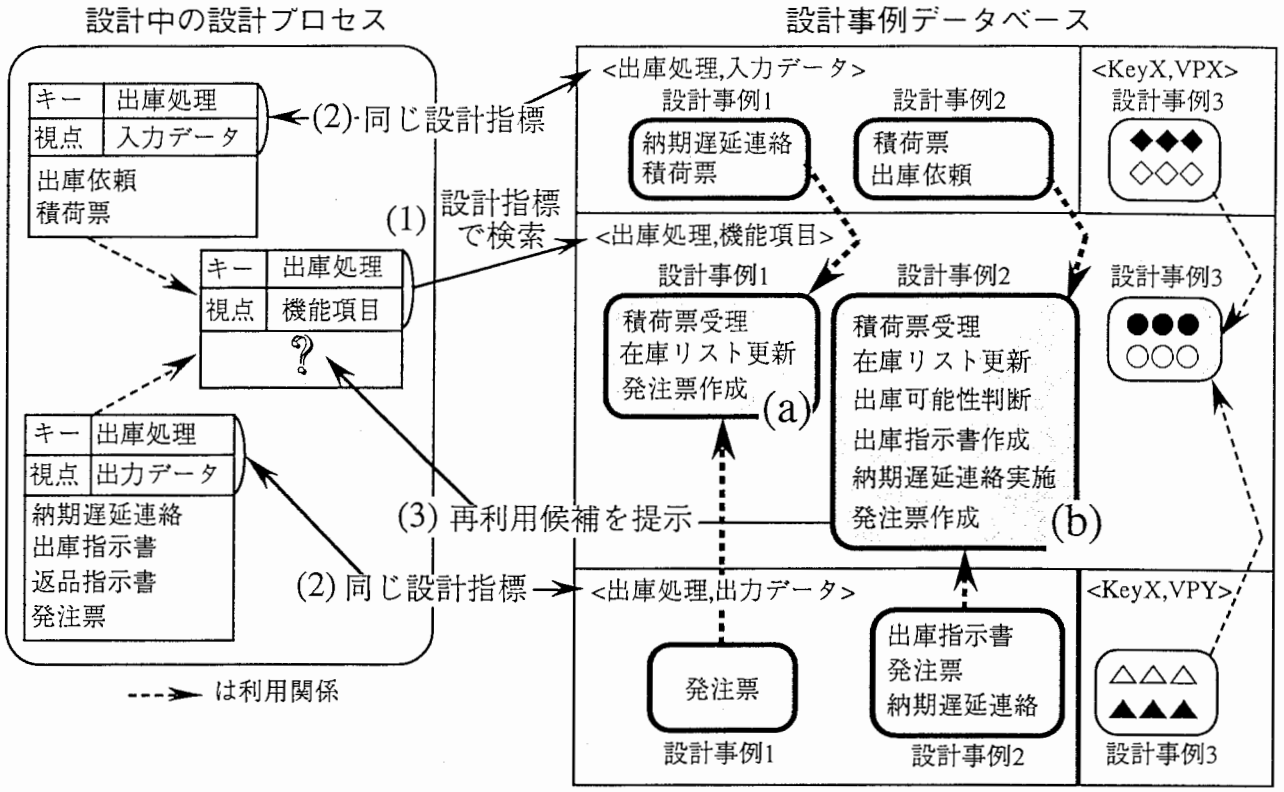

図 6　設計結果の提示方法

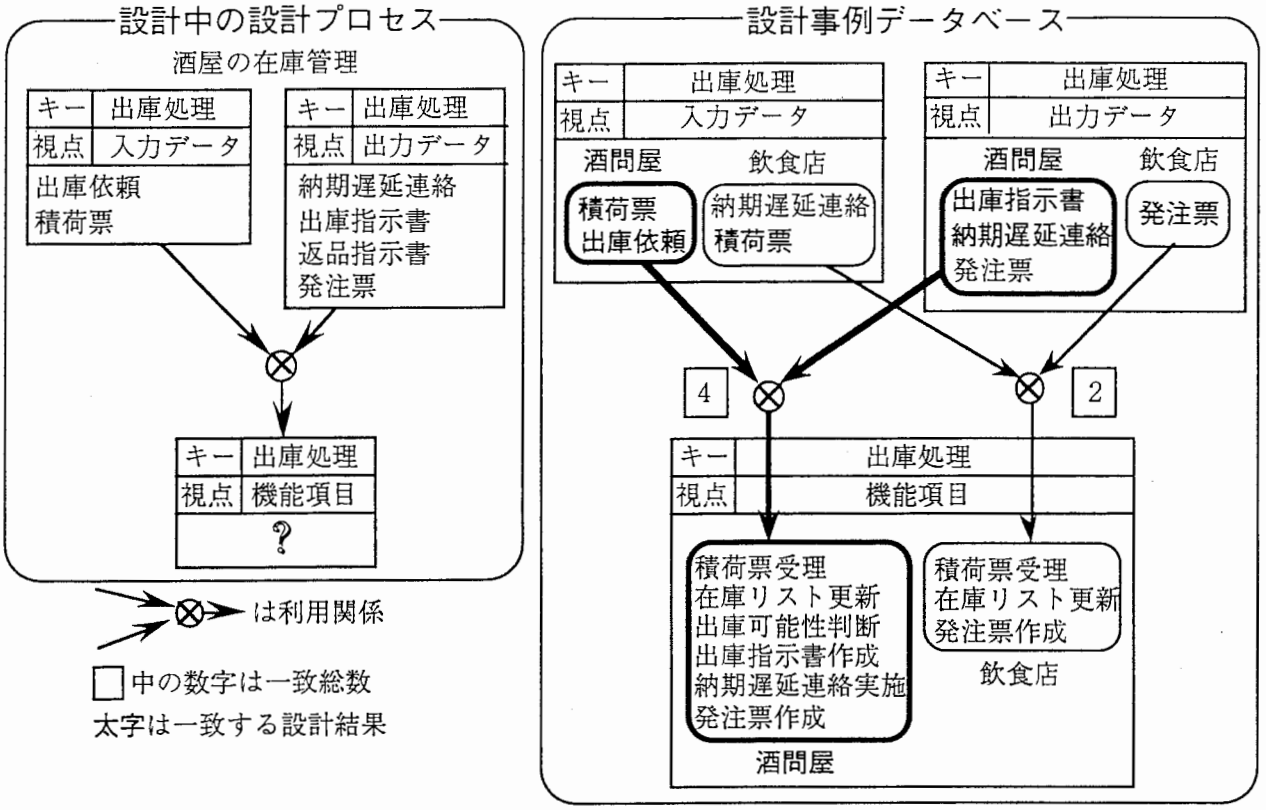

図7　一致総数による絞り込み

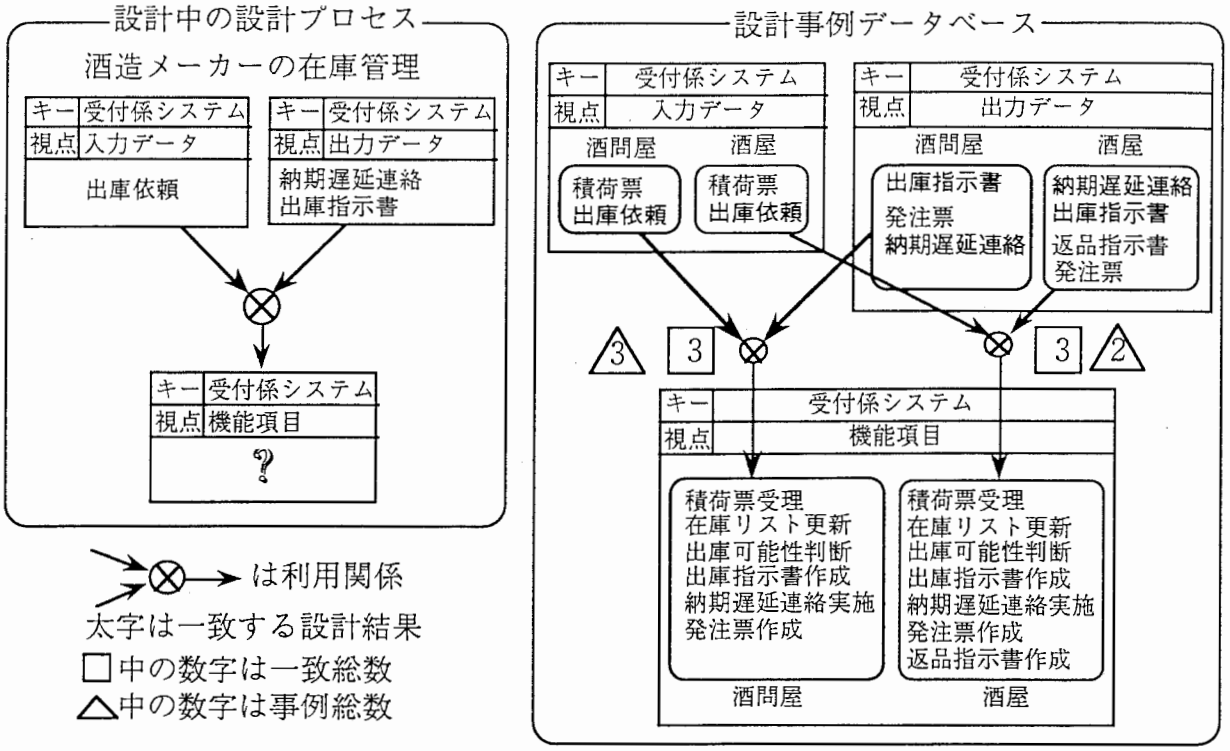

図８　事例総数による絞り込み

項目	ガイド有り	ガイドなし
総設計時間	20時間	24時間
設計ビュー数	139	109

表1　生産性の評価

※異なる設計問題（在庫管理領域の問題），被験者は同一

※実験は，ガイド有りDIGを先に行った

※設計ビュー数は設計の規模（問題の複雑さ）を表わす

設計手順の計算回数（設計目標の数）	44
設計手順の採用回数	17
再利用率	38.6%（17／44）

表2　設計方針の再利用性

総設計エンティティ数	88
再利用した設計エンティティ数	11
再利用率	12.5%（11／88）

表3　設計エンティティの再利用性

ドキュメント保守支援方式への適用

1. はじめに

ドキュメントの作成は比較的ソフトウェアの設計と類似しているため，本手法の適用による保守支援への効果が期待できる．以下では，本稿で提案した手法をドキュメント保守支援方式に適用するための方法について述べる．

ドキュメントには作成者が表わしたい概念体系が反映されている．ドキュメントの変更とはドキュメントに反映された概念体系の一部を変更することであり，その影響は変更された部分と関係を持つ全ての部分に波及する．

保守者がこのような影響波及解析作業を行うためには，作成者がドキュメントに表わそうとした概念体系を理解する必要がある．概念体系を理解するということは，ドキュメントを構成する文章／図／表が表わす概念，章や節のまとまりが表わす概念，さらにこれらがどのように関係しているのかを把握することであり，このような作業は，現状では保守者がドキュメントを読んで概念や関係を推測することによって行われることが多い．これは，ドキュメント作成時に考慮された概念や関係の情報がほとんど残されていないためであり，ドキュメントの変更に伴う影響波及解析作業を支援するためにはこれらの情報を記録することが不可欠であると考えられる．

我々はこれまで，，ソフトウェア設計を対象に設計時に設計プロセスを記録し影響波及解析で利用するための方法を提案するとともに[1][2]，提案した手法がソフトウェアの保守作業の負担軽減に有効であることを実験により明らかにした[3]．

我々は，提案した手法の考え方を取り入れ，ドキュメント作成時にドキュメントの変更に伴う影響波及解析で有効と考えられる情報を記録・利用することで，影響波及解析を容易化する方法の検討を行っている．具体的にはドキュメント作成時に構造情報（ドキュメントの章節…段落の階層構造およびドキュメント本体である文章／図／表），概念情報(文章／図／表とこれらをまとめたものである章節が表わす概念）およびこれらの間の関係を記録し，ドキュメントの変更に伴う影響波及解析作業で利用する．

概念情報を影響波及解析で用いることで，これまで保守者に要求されてきた概念体系の理解，変更部分と概念体系との対応の把握，概念体系の上で変更に対応する部分の推測といった作業の支援が可能となり，保守作業の負担軽減の効果が期待できると考えられる．

本論文では我々が検討している手法の基本的な考え方を示す．まず，記録すべき情報を示し(2章)，本手法を実現するための機能構成について述べる(3章)．次に情報の記録方法，記録した情報の利用方法の概要をそれぞれ示し(4章，5章)，最後に考察する(6章)．

2. 記録情報

本章では，最初にドキュメント作成作業およびドキュメント保守作業の分析を通じて，ドキュメントに反映されている概念体系と文章／図／表および章節の構造との関係について簡単な考察を行い，影響波及解析で有効と考えられる情報を明確にした後，作成時に記録すべき情報の内容を示す．

ドキュメント作成／保守作業

ドキュメントは作成者が表わしたい概念体系を反映していると考えられる．この概念体系は作成者が最初から体系として持っているわけではなく，ドキュメントを作成する過程で，ある観点に着目し作成者の頭の中に存在する複数の概念をまとまりとして捉える作業を繰り返すことによって構築されたものであると考えられる．例えば，図1①は「生物分類」という観点から「霊長類」，「ヒト」，「サル」をまとまりとして捉えたことを表わ

している．

そして，このように，ある観点から捉えた概念のまとまり（以下，概念集合と呼ぶ）が，ドキュメントの本体である文章／図／表として表わされていると考えられる．従って，ドキュメントに反映されている概念体系の実体はこれらの文章／図／表であると考えることができる．例えば，図１①を文章化したものが図１(a)である．

文章／図／表間の関係は，このような考え方に基づくと概念集合間の関係として捉えることができる．また，章節の構造と文章／図／表の関係は，章節を構成するものが文章／図／表であることから，文章／図／表が表わす概念集合を１つの概念とみなすと，これらの関係は概念集合そのものとして考えることができる．ここで，章節の見出し（もしくは，見出しに近い意味を表わす語）が概念集合における観点になると考えられる．例えば，図１では「共通点」という見出しがあり，これが生物分類上の共通点について書かれた文章や体の構成上の共通点について書かれた文章で構成されている．これは，「ヒト」と「サル」「生物分類の話」，「体の構成」を「共通点」という観点から捉えたことを表わしている．

このようにドキュメントを構成する文章／図／表や章節の構造とドキュメントに反映されている概念体系との対応を捉えると，ドキュメントの変更とは，「概念集合に含まれる概念」，「観点」，「概念集合間の関係」の何れかの変更として捉えることができる．そして，これらの変更による影響は，概念集合から構築された概念体系上関係を持つ他の概念集合に波及すると考えられ，その結果が文章／図／表または章節の構造への影響として現われる．

例えば，図１(b)で示されている文章の変更が，図１②で表わされている「サルにはコミュニケーション能力が無い」という概念のまとまりを「サルにはコミュニケーション能力がある」に変更することであった場合，図１ュの太枠部分と概念体系上の関係を持つすべての概念のまとまり（図１③）に影響が波及する可能性があり，そのため，図１③を表わす文章（図１(c)）が影響波及箇所となる．

このようなドキュメントの変更に伴う影響波及解析作業を整理すると，以下のような作業として捉えることができる．

(1) ドキュメントを読むことによってそのドキュメントがどのような概念体系を表わしているのかを把握する

(2) 変更される部分がその概念体系上のどの部分を表わすものであるのかを認識する

(3) (2) と関係を持つ概念集合を洗い出す

(4) (3) で洗い出した概念集合に影響が波及するか否かを判断する

(5) (4) で影響が波及する場合，概念集合を捉え直すとともにその変更部分に対応する文章／図／表または章節の構造を修正する

(6) (2) ～ (5) を影響がなくなるまで繰り返す

しかし，通常このような作業を行う上で必要な概念体系に関する情報は残されてないことが多く，保守者はドキュメントの概念体系を理解するために，ドキュメント作成の場合と同様に，ドキュメントを構成する章節ごとに，それらを構成する文章／図／表が表わす概念集合を把握し，組み立てることによって最終的にそのドキュメントではどのような概念体系が表わされているのかを把握し，その上で影響波及解析作業を行うという困難な作業を行わなければならない．

従って，影響波及解析作業を容易化するためには，ドキュメントを作成する際に作成者が作成時に捉えた概念集合と文章／図／表および章節の構造それぞれの情報とそれらの間の対応を記録しておくことが必要であると考えられる．

以下，これらの情報として記録すべき項目を示す．

記録情報の内容

記録すべき情報には，(1) 概念情報，(2) 構造情報，(3) 概念情報と構造情報の間の関係がある（図２）．

(1) 概念情報

概念集合および概念集合間の関係を概念情報と呼ぶ．概念情報として記録する内容は以下の通り．

(a) 概念集合

概念集合には１つの観点と複数の概念が含まれる．実際は，概念と概念の関係（例えば，図１①では霊長類とヒトはis-a関係で結ばれている）も存在するが，影響波及箇所をたどる作業で重要な情報は，文章／図／表が表わす概念集合に含まれる概念間の関係ではなく，概念集合間の関係であるため，本情報としては，以下の項目について記録するものとする．

・観点（図２①）

複数の概念をどのような観点から捉えたのかを観点名として記述する．

・概念（図２の楕円部分）

作成者が捉えた概念（のまとまり）を示す概念名を記述する．

(b) 概念集合間の関係（図２②）

概念集合間に継承等（例えば，文章間に参照関係があるといったケース）の関係が存在する場合，関係を持つ概念集合間の関係（概念集合間の関係は，厳密には概念集合の部分集合間の関係など様々なものがあるため，ここではこれらを総称して概念集合間の関係と呼ぶ．概念集合の詳細については考察で述べる）を記録する．

(2) 構造情報

本情報はドキュメントの構造（章節…段落といった階層構造．以下，階層情報と呼ぶ）とドキュメント本体（文章／図／表．以下，本体情報と呼ぶ）からなり，それぞれ以下のように記述する．

(a) 階層情報（図２③）

章節…段落（文章，図，表は段落の内容であると考える）の階層関係を記述する．階層情報としては，章節…段落の見出し，およびその章節を構成する下位の見出しを記述する．なお，階層構造の終端を段落と考え，下位の構造を持たない．

(b) 本体情報（図２④）

本体情報は，段落の見出し，文章／図／表で表わされる段落の内容そのものを記述する．

(c) 階層情報と本体情報間の関係（図２⑤）

階層情報で記録された見出しのうち，それが段落である場合は本体情報と階層情報の関係として記録する．

(3) 概念情報と構造情報間の関係（図２⑥）

概念情報と構造情報間の関係は，概念集合と階層情報の関係から求めることができる．従って，ドキュメント作成時には，概念集合と階層情報間の関係を記録する．

3. 機能構成

現在，本手法を実現するために必要な機能の検討を行っており，その機能構成を図３に示す．本手法を実現するための機能は大きく分けて，(1)記録支援機能，(2)保守支援機能の２つから構成される．以下，それぞれの機能概要を示す．

(1) 記録支援機能

記録支援機能は以下の機能から構成される．

(a) 概念情報記録支援機能

本機能は，概念情報の記録のためのエディタおよび記録を支援する機能から構成される．記録を支援するための機能として，記録した概念集合のブラウジングを行う機能，概念を表わす名称のバラツキを防止するための，類語辞書と類義判定機能，概念情報における「観点名」，「概念名」を記録する際に，その都度記述していたのでは記録の負担がかかるため，予め領域毎に概念のまとまりを記述した概念辞書を用意し，これを利用することで概念情報の記録を支援する機能の検討を行っている．概念名のバラツキ防止機能と概念辞書については考察で示す．

(b) 構造情報記録支援機能

本機能は，階層情報と本体情報をそれぞれ記録するためのエディタおよび記録した情報のブラウジングを行う機能からなる．階層情報を記録するためのエディタはテキストベースのものを用意し，本体情報を記録するためのエディタは，その表現形式に合わせ，テキストエディタ，グラフィックエディタ，表エディタの３種類を用意する．

(c) 記録情報の洩れを検出する機能

ドキュメントが完成した状態では，概念集合－本体情報－階層情報の３つが関係付けられて記録されていなければならない．本機能はこれらの情報の洩れを検出し，洩れている情報の記録を作成者に促す．

(2) 保守支援機能

本機能は，記録した情報を用いて，変更箇所と概念体系上の関係を持つ概念集合を洗い出し，保守者にその影響波及箇所を文章／図／表または章節のまとまりとして提示する．

4. 情報の記録方法

本章では，記録支援機能を用いた情報の記録イメージを示す．本来，構造情報，概念情報の記録の順序は不定であるため，以下では階層情報を記録したものから概念情報→本体情報の順に記録を行う例を示す（図４）．

(1) 作成者は階層情報として記録した見出しの１つを指定し，概念集合を記述するエディタを起動する

・作成者は「生物分類」を指定する（図４①）

(2) 作成者は概念集合を記述する

・作成者は概念集合として，観点「生物分類」，概念「霊長類，ヒト，サル」を記述(図４②)

・作成者が情報の記録終了をシステムに通知すると，システムは階層情報と概念集合との間の関係を自動的に記録する（図４❶）

(3) 作成者は階層情報として記録した見出しの１つを指定し，本体情報を記述するエディタを起動するとともに，エディタの種類を選択する

・作成者は「生物分類」を指定（図４③）し，エディタはテキストエディタを選択する

(4) 作成者は本体情報を記述する

・作成者は段落の内容を文章で記述する（図４④）

・作成者が情報の記録終了をシステムに通知すると，システムは階層情報と概念集合との間の関係を自動的に記録する（図４❷）

5. 影響波及解析方法

以下，変更に伴う影響波及解析方法の概要について述べる(図5).

(1) 保守者はドキュメントの変更対象部分（文章／図／表など）を修正する

・「生物分類から見たヒト」に関する文章で「サル」に関する記述が修正されたとする（図５①）

(2) 保守者は変更した文章と関係を持つ階層情報および概念集合を必要に応じて修正する

・システムは，変更された本体情報と関係を持つ階層情報の見出しを検索し保守者に提示する（図５②）

・保守者は提示された見出し（図５②）は変更の必要が無いと判断した

・システムは，見つかった階層情報の見出しと関係を持つ概念集合を検索し保守者に提示する（図５③）

・保守者は提示された概念集合を変更する

(3) 保守者は，変更した概念集合と関係を持つ概念集合を検索し必要があれば修正する

・システムは(2)で変更された概念集合と他の概念集合との関係をたどり，見つかった概念集合を保守者に提示する（図５④）

・保守者は，提示された概念集合を修正する

(4) 保守者は修正した概念集合に対応する階層情報および本体情報を必要に応じて修正する

・システムは変更した概念集合と階層情報の関係をたどり，見つかった階層情報の見出しを提示する（図５⑤）

・保守者は提示された見出し（図５⑤）は変更の必要が無いと判断した

・システムは見つかった階層情報の見出しと関係を持つ本体情報を検索し，見つかった本体情報を保守者に提示する（図５⑥）

・保守者は，提示された本体情報を修正する

(5) 以下，(1) ～ (4) を影響波及解析対象が無くなるまで繰り返す.

5. 考察

本稿では，ドキュメントが変更された場合の影響波及解析を支援するために「概念情報」と「構造情報」を記録することが有効であることを示し，これらの情報を利用することで影響波及解析を容易化する方法を提案した．概念情報を用いることにより，影響波及箇所の洩れを検出するとともに，影響波及箇所の絞り込みが可能になり，ドキュメント保守工数の削減が期待できる.

今後は，以下の課題について検討を進めていく予定である.

(1) 概念情報の記録形式

概念情報の記録に関して，以下の課題がある.

(b) 概念集合間の関係の記述

概念集合間の関係には様々なパターンがある（図６に一例を示す）．例えば，図６(a)の「Y」の関係は，概念集合（X,Y,Z）の概念（Y）と概念集合（Y,P,Q）の概念（Y）との関係を表わしており，図６(b)の場合は概念集合（X,Y,Z）の部分集合（X,Y）と概念集合（Y,P,Q）の概念（Y）との関係を表わしている．また，図６(c)のように概念集合(X,Y,Z)と概念集合（Y,P,Q）の概念（Y）との関係というパターンもある．これ以外にも多数の

関係のパターンが考えられるが,特に部分集合や集合全体との関係の場合は,部分集合（X,Y）と概念（Y）との関係の例のように概念（Y）を新たに定義することであると考えられる．従って，このような関係を記述するための記述形式の検討を行う必要がある．

(2) 概念名のバラツキの吸収

本方式では作成者が任意に概念名を付加しそれらのまとまりを概念集合として記述する．この時，概念名のバラツキの問題が生じる．概念名のバラツキは影響波及解析時の影響波及箇所の検索効率に影響を与える．現在，このような問題を解決するための方法として類義語辞書および概念辞書の活用を考えている．

類義語辞書は既存の類語辞書もしくはユーザ定義の類語を登録するとともに，品詞，活用，単数／複数の相違も既存の辞書を流用することで吸収しようと考えている．

概念辞書は概念名の使用をドキュメント内で一意に保つことを目的としたもので，予め作成しておく．

(3) 記録の負担の軽減

本方式では，従来記述していた情報（本手法では構造情報に相当する）に加え，章節…段落の見出し毎に概念のまとまりやその間の関係を記録する必要がある．このような作業は，一般に非常に多くの情報の記述を作成者に課すことになるため，作成者の負担が大きい．

このような作業を支援するための方法として，予め領域毎に概念構造を記述した概念辞書を用意し，領域に共通の概念については，それを参照することにより記述の負担の軽減を行うことを考えている．

(4) 既存のドキュメントの保守支援

新規に作成するドキュメントについては，ドキュメント作成時に構造情報および概念情報を記録することによる支援が期待できるが，既存のドキュメントに対しては，その文章を読んで理解した上で概念情報を記述するという作業になってしまう．このような作業を支援するための方法として自然言語処理の応用[4][5]が考えられる．例えば，情報検索で用いられている自然言語処理や，機械翻訳で用いられている自然言語処理は，その応用が期待できると考えられる．

【参考文献】

1. 浜田,安達,竹中,"設計プロセスの蓄積・利用による設計支援法について",情報処理学会ソフトウェア工学研究会,1991
2. 安達,浜田,竹中,"設計プロセスを利用した修正支援法について",情報処理学会ソフトウェア工学研究会,1991
3. 浜田,安達,"設計プロセスを利用したソフトウェア修正支援方式",情報処理学会　論文誌,Vol.35 No.5,1994
4. 藤澤,絹川,"情報検索における自然言語処理",情報処理学会　学会誌,Vol.34 No.10,1993
5. 江原，田中,"機械翻訳における自然言語処理",情報処理学会　学会誌,Vol.34 No.10,1993

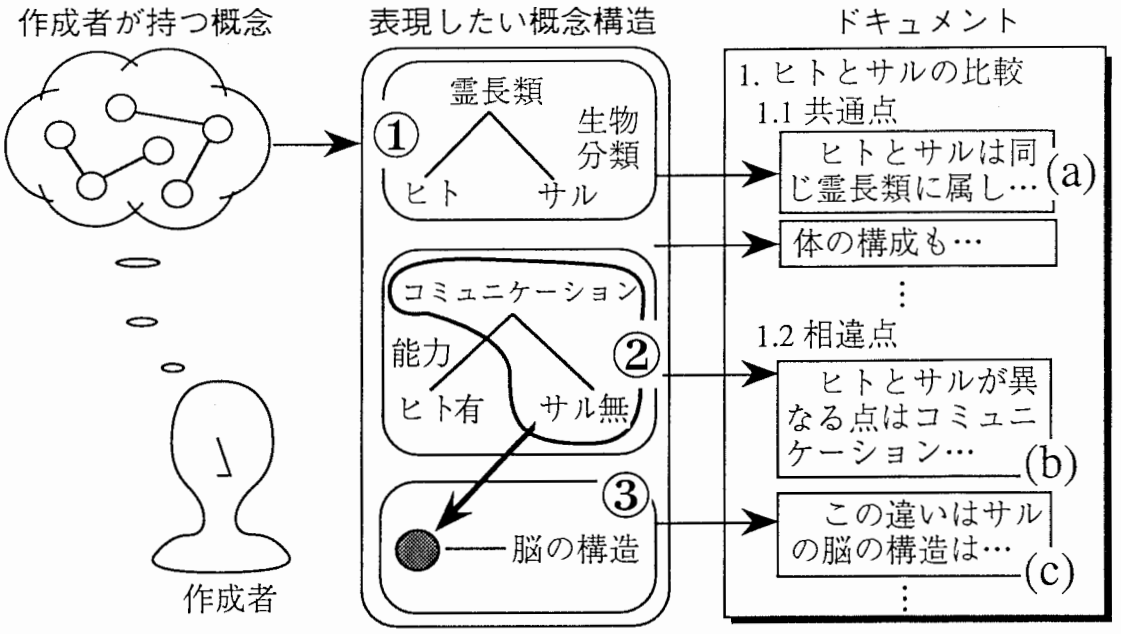

図1　ドキュメント作成の流れ

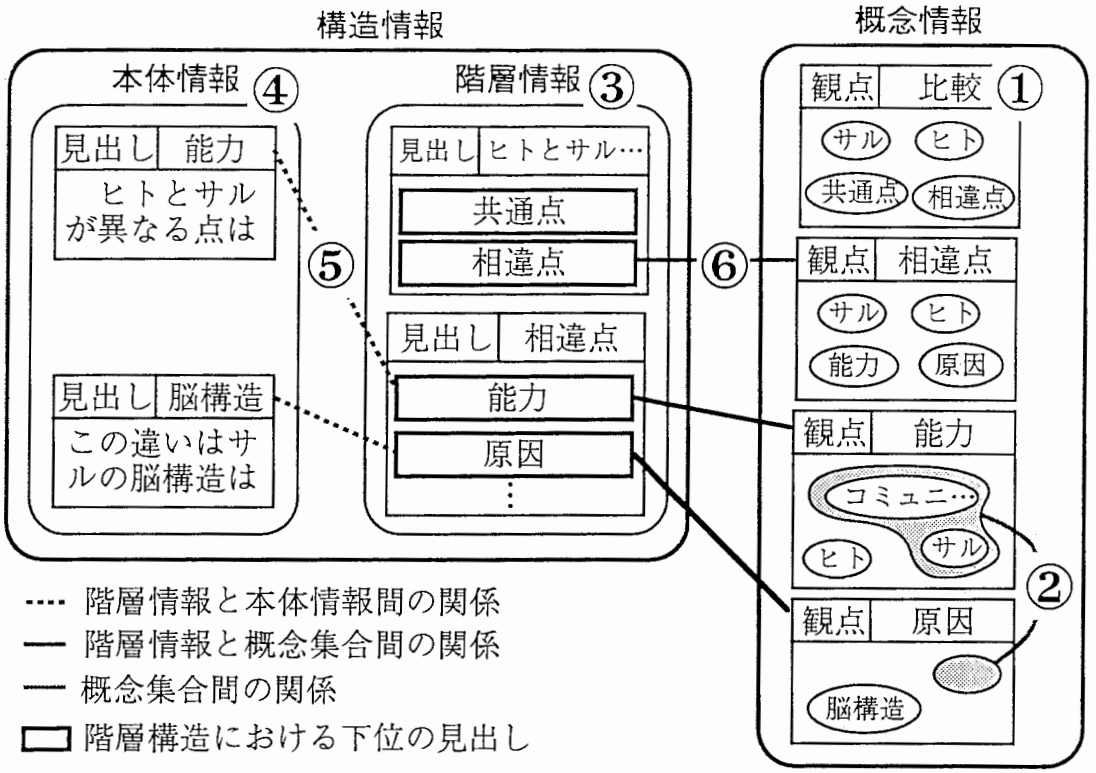
構造情報
概念情報
本体情報 ④
階層情報 ③
見出し 能力
ヒトとサルが異なる点は
見出し 脳構造
この違いはサルの脳構造は
⑤
見出し ヒトとサル…
共通点
相違点
⑥
見出し 相違点
能力
原因
観点 比較 ①
サル
ヒト
共通点
相違点
観点 相違点
サル
ヒト
能力
原因
観点 能力
コミュニ…
サル
ヒト
②
観点 原因
脳構造
階層情報と本体情報間の関係
階層情報と概念集合間の関係
概念集合間の関係
階層構造における下位の見出し

図２　記録情報の例

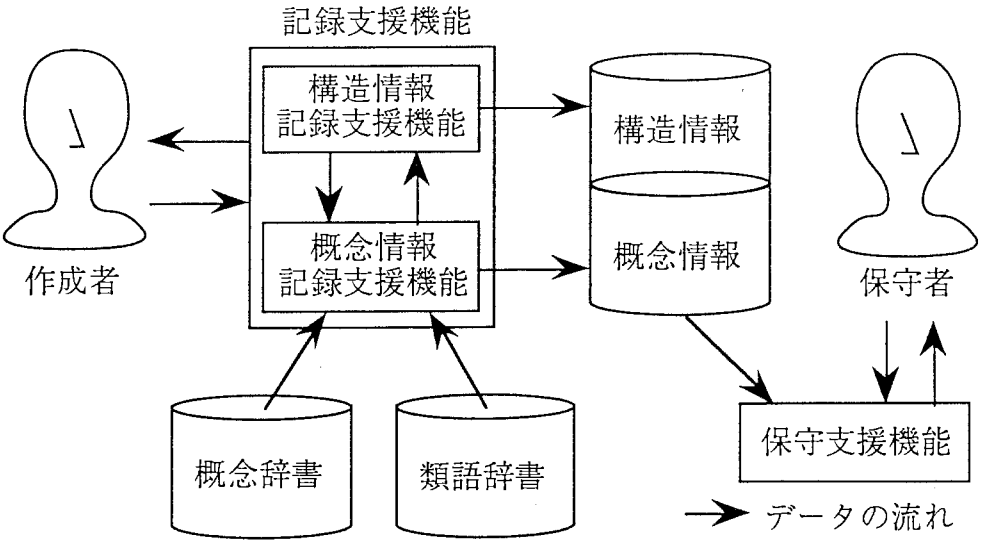
記録支援機能
構造情報
記録支援機能
概念情報
記録支援機能
構造情報
概念情報
作成者
保守者
概念辞書
類語辞書
保守支援機能
データの流れ

図3　システム構成

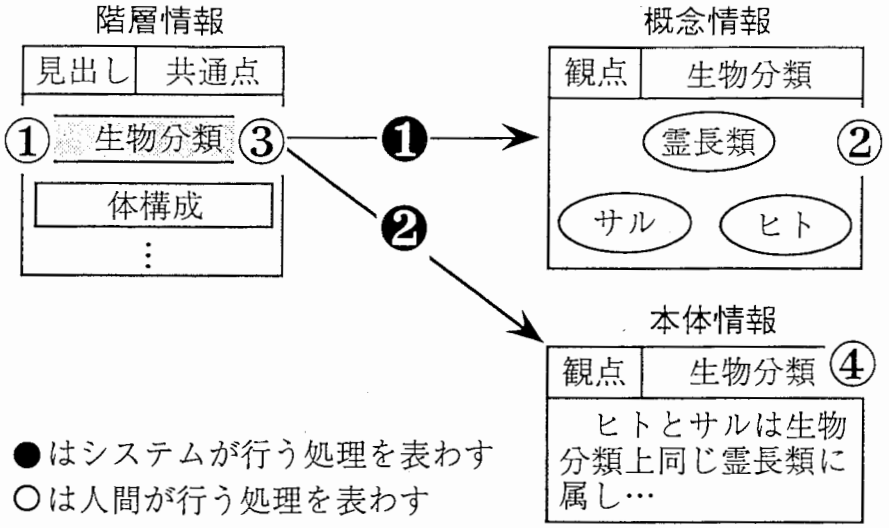
階層情報
見出し
共通点
①
生物分類
③
体構成
概念情報
観点
生物分類
霊長類
②
サル
ヒト
本体情報
観点
生物分類
④
ヒトとサルは生物分類上同じ霊長類に属し…
●はシステムが行う処理を表わす
○は人間が行う処理を表わす

図4　記録のイメージ

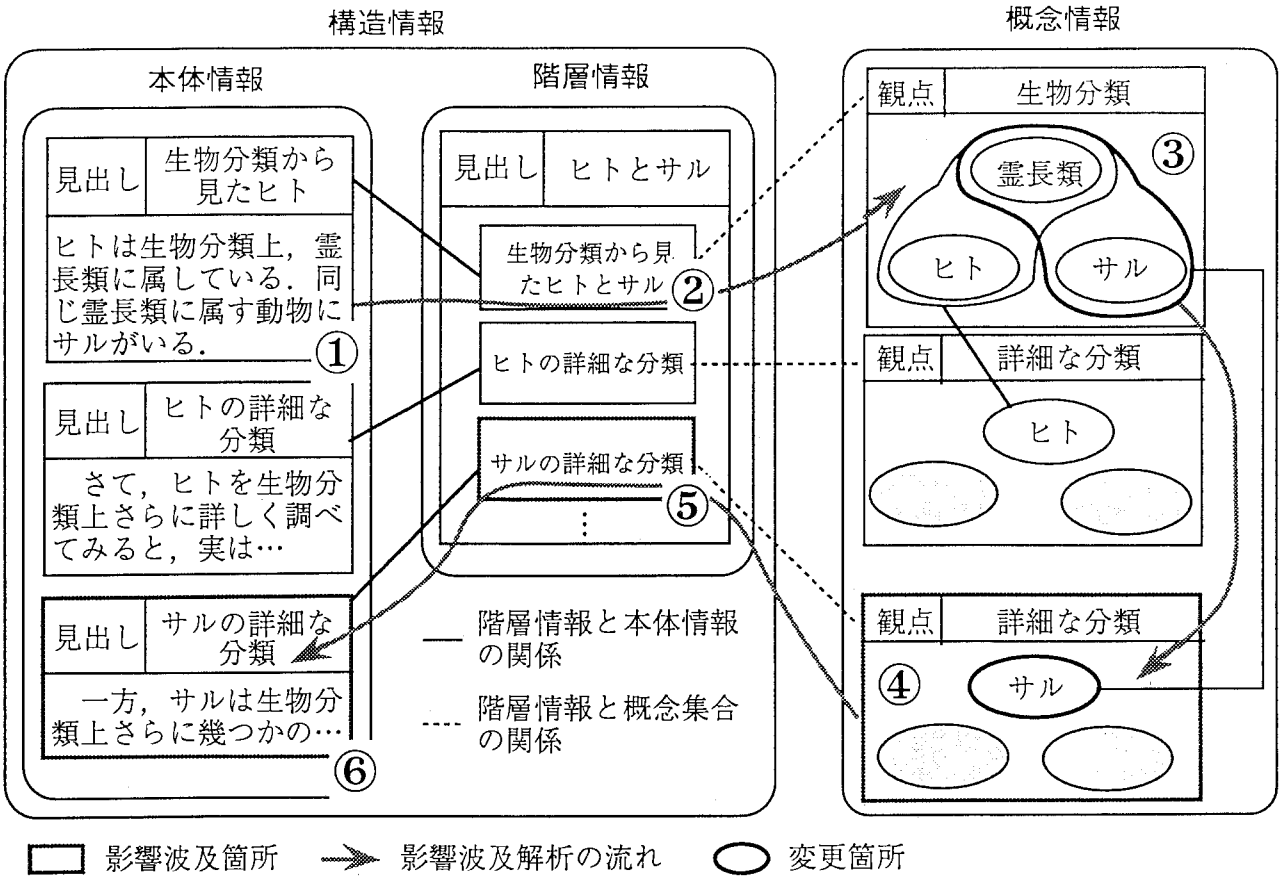
構造情報
概念情報
本体情報
階層情報
見出し
生物分類から見たヒト
ヒトは生物分類上，霊長類に属している．同じ霊長類に属す動物にサルがいる．
①
見出し
ヒトの詳細な分類
さて，ヒトを生物分類上さらに詳しく調べてみると，実は…
見出し
サルの詳細な分類
一方，サルは生物分類上さらに幾つかの…
⑥
見出し
ヒトとサル
生物分類から見たヒトとサル
②
ヒトの詳細な分類
サルの詳細な分類
⑤
階層情報と本体情報の関係
階層情報と概念集合の関係
観点
生物分類
霊長類
③
ヒト
サル
観点
詳細な分類
ヒト
観点
詳細な分類
④
サル
影響波及箇所
影響波及解析の流れ
変更箇所

図５　影響波及解析方法

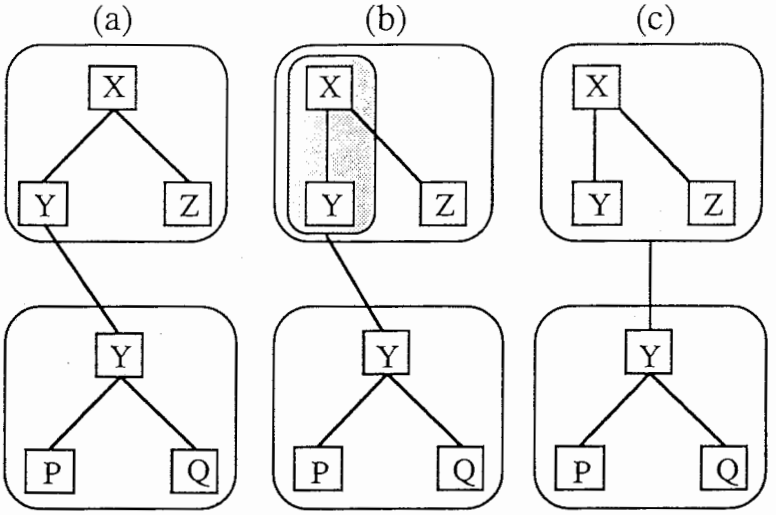

図 6　概念情報間の関係例

視点利用規則の記述形式

視点利用規則は，設計法や作業標準毎にユーザが自由に記述しても良いことになっている．視点利用規則の記述形式はprologの記述形式に準拠している(若干の制約がある)が，特にprologの知識が無くても記述できるように幾つかの予約語を用意している．以下では，最初に視点利用規則の書式について述べ，次に各予約語の書式および意味の説明を行う．また，幾つかの予約語を用意しこれを用いることで

規則の書式

視点利用規則は以下の書式で記述する．

<ルールインデックス>:-

<設計生産物の条件>のならび．

左辺:<ルールインデックス>

どの設計法に関する規則なのかを表す．必ず記述する(書式は後述).

右辺:<設計生産物の条件>

<設計生産物が満たすべき条件>または<設計目標の提案>から構成される．以下にそれぞれの意味と、記述上の制約について示す．

<設計生産物が満たすべき条件>:

・設計すべき設計生産物が設計できるための条件を記述する(条件の種類と書式については後述する)

・0個以上記述する(<設計目標の提案>のみの規則もあり得る)

・<設計生産物が満たすべき条件>は、それが成立した時に設計すべき<設計目標の提案>より左側に書かなければならない

<設計目標の提案>:

・設計すべき設計生産物を記述する(設計目標の種類と書式については後述する)

・設計生産物はキーと視点のペアで表現する

・1個以上記述する

上記の形式で記述された視点利用規則は、「<設計目標の提案>の左側に書かれた<設計生産物が満たすべき条件>が全て満たされれば<設計目標の提案>を行う．このような設計目標を全て求める」という意味を表す．

ルールインデックス

どんな設計法についての規則であるのかを表す．1つの設計法について1つのルールファイル(ルールグループと呼ぶ)を作成する．将来的には、ルールグループを結合し設計の一連の流れを表すルールグループとして利用することも考えている．

書式:

vr (ルールグループ名)

ルールグループ名はアトム．

記述例:

vr ('SA') :-

designView (Key , 原要求) , design (Key , 端末) .

※SA は「要求分析工程の設計に関するルールグループ名」という意味で付けたもの.

設計生産物が満たすべき条件の種類と書式

設計目標が設計できるために設計されていなければならない"設計生産物の存在"に関する条件、"設計生産物間の設計上の利用関係"に関する条件、キー、視点、設計エンティティの指定を行うためのものがある.

(1) 原要求が存在する

書式 : requirement (キー , 原要求)

項のうち、キーは定数または変数.

項のうち、"原要求"は定数.

未設定の変数はユニフィケーションされる.

記述例 :

requirement (X店在庫管理 , 原要求)

X店在庫管理についての原要求が存在する

requirement (Key , 原要求)

(ある) Key についての原要求が存在する

(2) (ある1つの) 設計生産物が存在する

書式 : designView (キー , 視点)

項は定数または変数.

未設定の変数はユニフィケーションされる.

記述例 :

designView ('X店在庫管理' , プロセス構成)

設計ビュー <X店在庫管理 , プロセス構成> がある

designView (Key , プロセス構成)

視点 "プロセス構成" を持つ設計ビューがある

※キーはユニフィケーションされる

(3) (全ての) 設計生産物が存在する

書式 : allDesignViews (キーのリスト , 視点のリスト)

項は定数または変数.

未設定の変数はユニフィケーションされる.

記述例 :

allDesignViews ([入庫処理 , 出庫処理] , [プロセス間のデータ入出力])

<入庫処理,プロセス間のデータ入出力> , <出庫処理,プロセスの間データ入出力> が全てある.

allDesignViews (KeyLists , [プロセス構成 , データ構成])

視点 "プロセス構成" と "データ構成" を持つ設計ビューのうちキーが共通している設計ビューを全て求める.

※KeyListsがユニフィケーションされる

(4) (ある1つの設計エンティティまで指定したーある1つの) 設計生産物が存在する

書式 : designEntity (キー , 視点 , Entity)

項は定数または変数.

未設定の変数はユニフィケーションされる.

記述例 :

designEntity ('X店在庫管理' , プロセス構成 , 'X店在庫管理システム')

キー "X店在庫管理" , 視点 "プロセス構成" を持つ設計ビューがある.

<X店在庫管理 , プロセス構成> が設計エンティティ "X店在庫管理システム" を持つ.

designEntity (Key , プロセス構成 , 'X店在庫管理システム')

視点 "プロセス構成" を持つ設計ビューがある.

※Key はユニフィケーションされる ("X店在庫管理" だったとする).

<'X店在庫管理' , プロセス構成> という設計ビューに設計エンティティ "X店在庫管理システム" を持つものがある.

※Key が確定する.

(5) (ある設計エンティティの集合まで指定したーある1つの) 設計生産物が存在する

書式 : allDesignEntities (キー , 視点 , 設計エンティティのリスト)

項は定数または変数.

未設定の変数はユニフィケーションされる.

記述例 :

allDesignEntities ('X店在庫管理システム' , プロセス構成 , [入庫処理 , 出庫処理])

キー "X店在庫管理システム" , 視点 "プロセス構成" を持つ設計ビューがある.

<X店在庫管理システム , プロセス構成> が設計エンティティ"[入庫処理 , 出庫処理]" を持つ.

allDesignEntities (Key , プロセス構成 , [入庫処理 , 出庫処理])

視点 "プロセス構成" を持つ設計ビューがある.

※Key はユニフィケーションされる ("X店在庫管理システム" だったとする).

<X店在庫管理システム , プロセス構成> が設計エンティティ "[入庫処理 , 出庫処理]" を持つ.

※Key が確定する.

設計目標の提案の種類と書式

設計すべき設計生産物をキーと視点で表す. 単に設計する設計生産物を表すだけでなく、設計の型 (全ての、1つの、どれか1つ) を表すことができる.

(1) (ある1つの)設計生産物を設計する

書式 : design (キー , 視点)

項は定数または変数.

変数はユニフィケーションされていなければならない. 従って、意味は以下のケース

と全て同じ.

記述例 :

design ('X店在庫管理' , プロセス構成)

<X店在庫管理 , プロセス構成> を設計する.

(2) (全ての) 設計生産物を設計する

書式 : designAll (キーのリスト , 視点のリスト)

項は定数または変数.

変数はユニフィケーションされていなければならない. 従って、意味は以下のケースと全て同じ.

記述例 :

design ([入庫処理,出庫処理],[プロセスへの入力データ,プロセスからの出力データ])

<入庫処理,プロセスへの入力データ>,<入庫処理,プロセスからの出力データ> ,

<出庫処理,プロセスへの入力データ>,<出庫処理,プロセスからの出力データ>

を設計する.

(3) (ある複数の) 設計生産物の中の、どれか1つを設計する

書式 : designAlt (キーのリスト , 視点のリスト)

設計生産物は、[キー , 視点] で表す.

項は定数または変数.

変数はユニフィケーションされていなければならない. 従って、意味は以下のケースと全て同じ.

記述例 :

designAlt ([[X店在庫管理システム,'MS'],[X店在庫管理システム,詳細化プロセス]])

<X店在庫管理システム,MS> を設計するか、<X店在庫管理システム,詳細化プロセス> を設計する.

(4) 設計の終了を表す語

書式 : end

end がキーまたは視点として記述されている場合には、設計 (詳細化) を終了する.

付録３

知識工学分野における本研究の位置付け

－　事例ベース推論技術の応用　－

知識工学の分野ではエキスパートシステムに代表される知識を用いた問題解決のための研究が行われている．本資料では，知識工学で行われている知識活用の研究について簡単に示した後，知識工学の分野における本研究の位置付けを示す．

知識工学における知識獲得の問題点

知識工学の分野では，専門家の知識を利用した問題解決を行うためのシステムとしてエキスパートシステムの研究が行われている．エキスパートシステムは専門家の問題解決作業を分析し問題解決に必要な知識を抽出し，これをルール形式で記述したものを用いて，専門的な知識を持たない者でも問題解決が行えるようにすることを目指したものである．しかし，専門家の知識を抽出することは非常に困難で，知識工学の分野ではこれを『知識獲得のボトルネック』と呼んでいる（図１）．

事例ベース推論

このような問題に対し，事例ベース推論（CBR : Case Based Reasoning）と呼ばれる，専門家が問題解決を行う際に過去の事例を積極的に活用していることに着目し，問題解決の過程を記録しておき，新たな問題解決を行う際に，その問題と類似している過去の事例を検索し活用する研究が行われている（図２）．事例ベース推論は以下のような推論方式を採る[1]（図３）．

(1) 過去に経験した事例（問題解決の過程）を事例ベースに蓄積する

(2) 新たな問題を解決する時に，その問題に類似した過去の問題を事例ベースから検索する

(3) 過去の問題に対する解決の事例を利用することによって当該問題を解決する

(4) 当該問題をその解答や反省材料とともに事例ベースに格納する

事例ベース推論は専門家の問題解決の過程を記録するため，『知識獲得のボトルネック』の問題は発生しないと考えられる．しかし，事例の蓄積方法等の課題も存在する．

関連研究

知識工学では，事例ベース推論の他にモデルベース推論，ルールベース推論といった方法が研究されている[2]（図４）．以下，各推論方式について簡単に述べる．

モデルベース推論（MBR : Model Based Reasoning）

モデルベース推論は一般法則（物理法則など）に基づく推論方式で，適用範囲が広いが知識獲得が非常に困難である．

ルールベース推論（RBR : Rule Based Reasoning）

ルールベース推論は，エキスパートシステムが基づいている推論方式で，専門家の経験則をルール化し，それを用いて推論を行う．知識獲得は前述のように困難である．

本稿で提案する手法の位置付け

前述のように，本手法では設計手順（設計目標，設計指標）と設計内容の提案を行う．設計目標の計算で用いる視点利用規則は設計法などから抽出するものであり，ルールベース推論の考え方に基づくものであると言える．一方，設計指標および設計内容は当該

設計と類似していると考えられる過去の設計事例を提示することから，事例ベース推論の考え方に基づくものであるといえる．従って，提案する手法はルールベース推論と事例ベース推論の２つの推論方式を融合した手法であると言える．

参考文献

[1] 推論技術の観点からみた事例に基づく推論,松原,人工知能学会誌,pp.567-575,vol7,No.4,1992

[2] 事例ベース推論の現状と展望,小林,人工知能学会誌,pp.559-566,vol7,No.4,1992

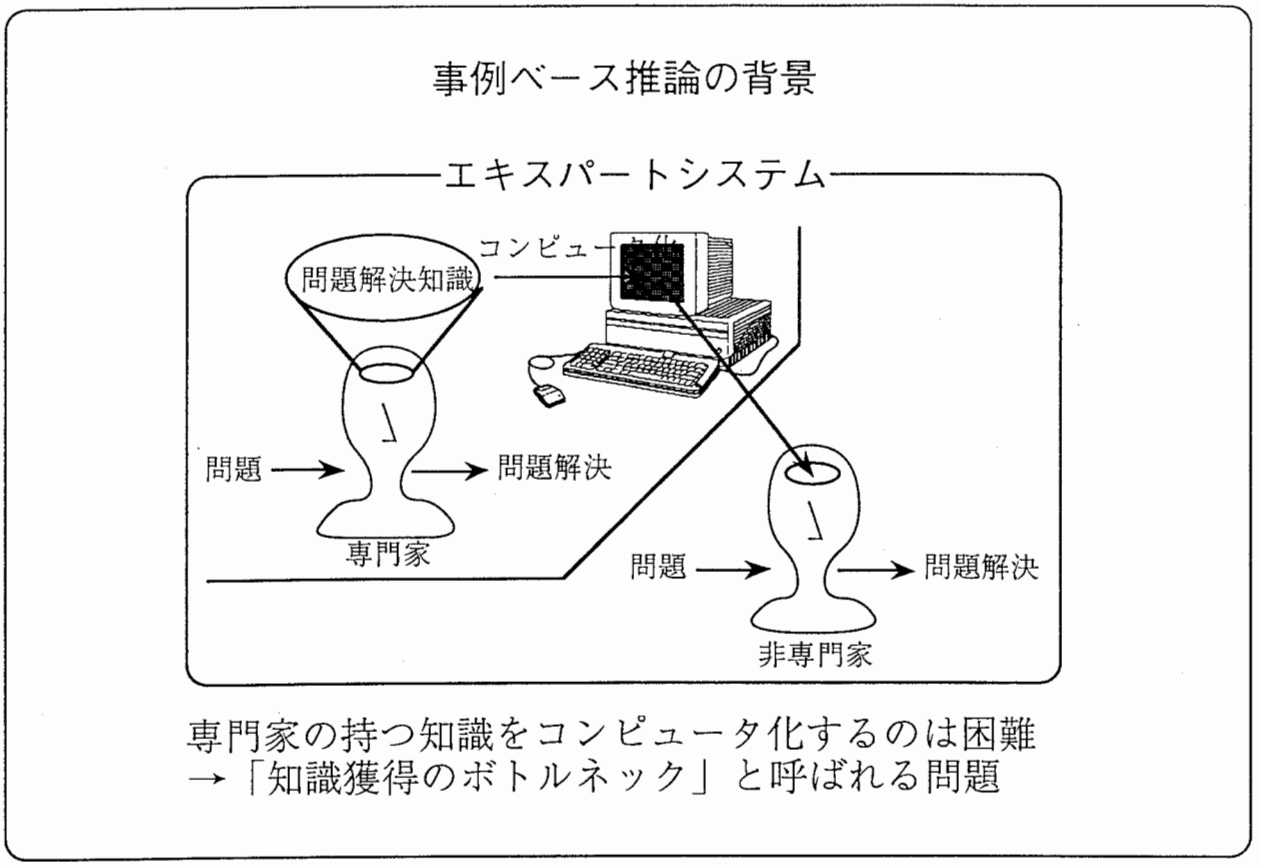

図１　事例ベース推論の背景

事例ベース推論（Case-Based Reasoning : CBR）

定義

与えられた問題に類似する過去の事例を直接利用して解を導く枠組み

説明

専門家は、新しい問題を解くのに、過去に類似の問題をどのように解決したかという事例や前例を参考にしていることが多い.

↓

専門家の思考パターンをコンピュータ化する

図2　事例ベース推論

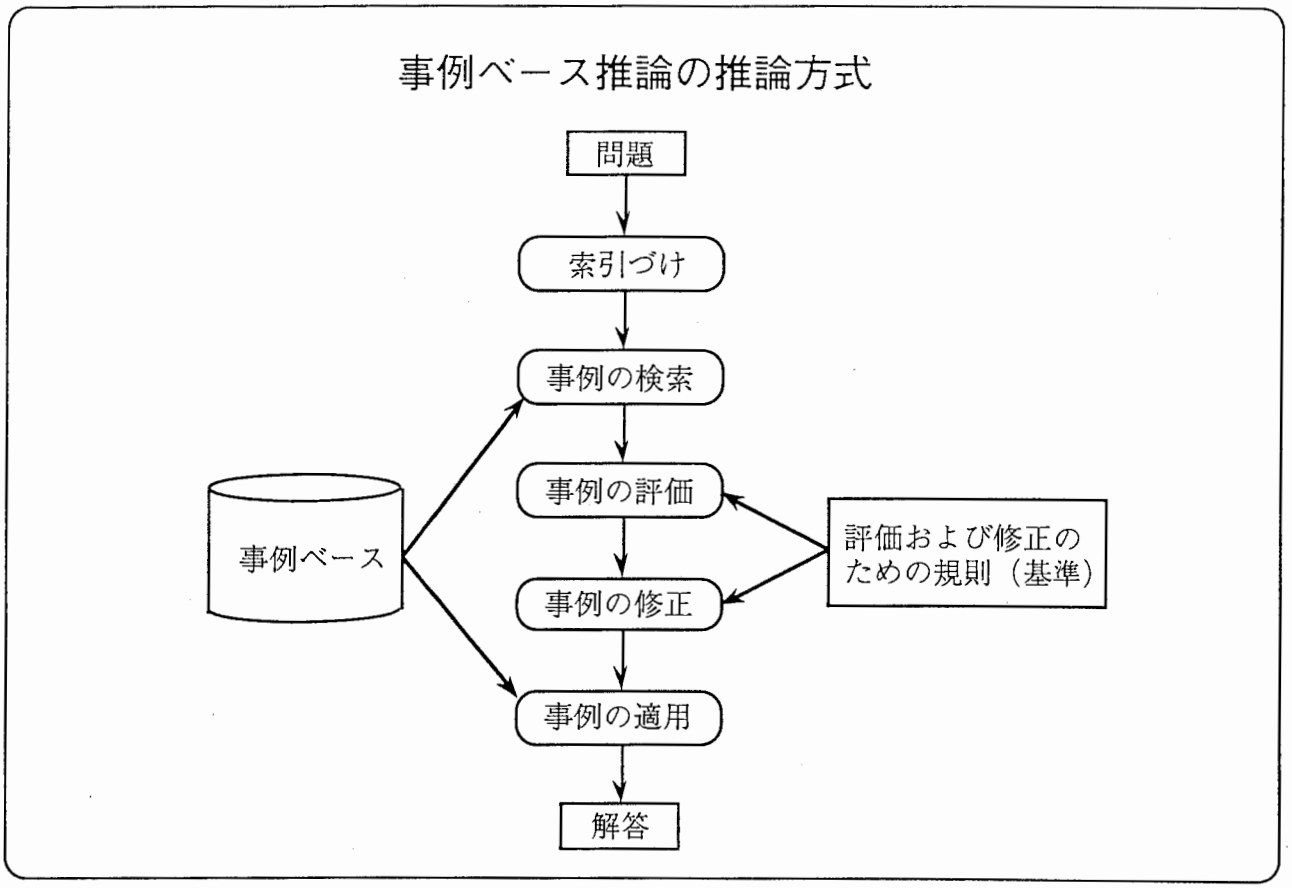

図３　事例ベース推論の推論方式

関連研究（その他の推論方式）

モデルベース推論 (Model-Based Reasoning : MBR)

対象とするシステムの構造や構成要素の特性に関する原理・原則的な知識に基づく推論

ルールベース推論 (Rule-Based Reasoning : RBR)

領域専門家の経験的知識に基づく推論

	MBR	RBR	CBR
何に基づく推論か？	原理・原則	経験的知識	事例
表現形式	ルール形式	ルール形式	フレーム形式
一貫性（無矛盾性？）	○	△	×
情報獲得の容易性	×	△	○

図４　その他の推論方式

ソフトウェアの設計支援への適用

従来の設計支援方式

プロセスプログラミングの考え方に基づく設計支援

・設計プロセスを予め定式化し利用する

→定式化できる範囲が狭い（設計法レベル）

事例ベース推論を用いた設計支援方式

・設計者の思考過程（設計情報）を記録・蓄積する

・新規設計を行う際に類似事例を検索し利用する

図5　ソフトウェア設計支援への適用