



Title	設計プロセスにおける仮説生成検証の動的展開に着目した設計支援フレームワーク
Author(s)	野間口, 大; 藤田, 喜久雄
Citation	人工知能学会論文誌. 2010, 25(3), p. 514-529
Version Type	VoR
URL	https://hdl.handle.net/11094/3064
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

設計プロセスにおける仮説生成検証の動的展開に着目した設計支援フレームワーク

A Design Support Framework through Dynamic Deployment of Hypothesis and Verification in the Design Process

野間口 大
Yutaka Nomaguch

大阪大学大学院工学研究科機械工学専攻

Department of Mechanical Engineering, Osaka University

noma@mech.eng.osaka-u.ac.jp, <http://syd.mech.eng.osaka-u.ac.jp/~noma/>

藤田 喜久雄
Kikuo Fujita

(同 上)

fujita@mech.eng.osaka-u.ac.jp, <http://syd.mech.eng.osaka-u.ac.jp/~fujita/>

keywords: knowledge management, hypothesis and verification process, design rationale, design support system, ontology

Summary

This paper proposes a design support framework, named DRIFT (Design Rationale Integration Framework of Three layers), which dynamically captures and manages hypothesis and verification in the design process. A core of DRIFT is a three-layered design process model of action, model operation and argumentation. This model integrates various design support tools and captures design operations performed on them. Action level captures the sequence of design operations. Model operation level captures the transition of design states, which records a design snapshot over design tools. Argumentation level captures the process of setting problems and alternatives. The linkage of three levels enables to automatically and efficiently capture and manage iterative hypothesis and verification processes through design operations over design tools. In DRIFT, such a linkage is extracted through the templates of design operations, which are extracted from the patterns embeded in design tools such as Design-For-X (DFX) approaches, and design tools are integrated through ontology-based representation of design concepts. An argumentation model, gIBIS (graphical Issue-Based Information System), is used for representing dependencies among problems and alternatives. A mechanism of TMS (Truth Maintenance System) is used for managing multiple hypothetical design stages. This paper also demonstrates a prototype implementation of DRIFT and its application to a simple design problem. Further, it is concluded with discussion of some future issues.

1. はじめに

設計は典型的な悪構造問題である。設計解に到達するまでの道筋が明確でないだけでなく、設計案を評価する際にも、与条件であるべき要求が不明確であり、何が良い解であるかどうか不明である [富山 02]。このため、設計案だけでなく解くべき問題自体も仮説であり、仮説の下で設計案を具体化して何らかの検証を行い、必要なら仮説を切り替えて別の設計案を設定することが不可欠であり、ある種の試行錯誤的なプロセスを通じて問題と解を同時に具体化していくことが本質となっている [Schön 82]。高度な設計支援を実現するためには、そのような仮説生成検証過程を形式化した上で、仮説の動的な切り替えなどの、従来は設計者の思考の中で暗黙的に行われていた内容を計算機上で明示的に取り扱うための何らかの枠組みを導入することが必要である。

本研究では、設計プロセスにおける仮説生成検証の動的な展開に着目した設計支援フレームワークとして DRIFT

(Design Rationale Integration Framework of Three layers) を新たに提案する。このフレームワークは仮説生成検証過程を設計対象表現の遷移とそれを駆動する議論すなわち問題の提起と代替案の生成や比較検討の連鎖として理解することを前提としており、その特徴は下記の 4 点に集約される。

- (1) 様々な問題と代替案、それらの依存関係における構造を獲得するために、設計操作のすべてを記録する。
- (2) 設計者が駆動する設計作業の副産物として (1) の記録と管理を行うための基盤である、行為・モデル操作・議論の 3 層統合プロセスモデルを導入する。
- (3) ある問題に対して代替案を生成する作業単位を設計操作のテンプレートとして定義しておき、それらを介して設計を進めることにより、(1) の内容を (2) のモデルに展開するようにする。
- (4) (3) により管理される様々な内容を設計対象概念のオントロジーに基づいた表現により統合する。

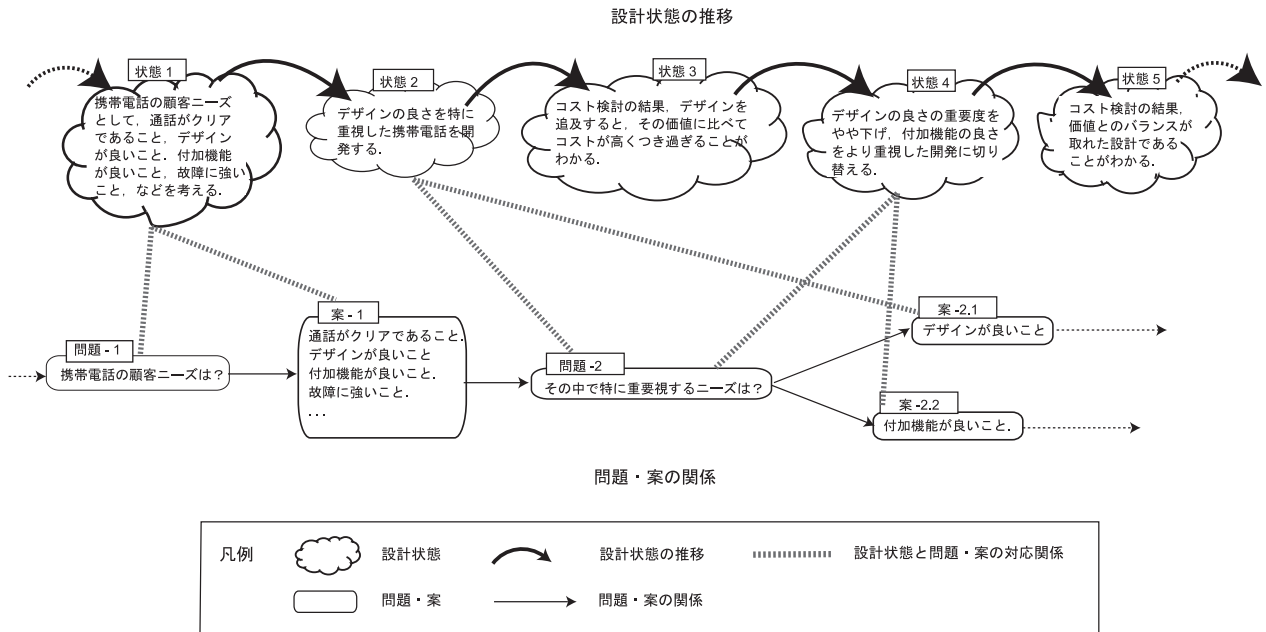


図1 設計状態の変遷と問題・案の変遷の例

本研究では、さらに、上記のフレームワークの実現性を示すために、(3)および(4)の内容について、典型的な DFX (Design For X) 手法 [Gatenby 90] を構成する操作の形式化とオントロジーによる表現を例示し、種々の設計手法がその規範となることを論じる。フレームワークの実現に際して、議論の連鎖を表現するために gIBIS [Conklin 88] の形式を、また、仮説の動的な切り替えのために真理性管理機構 (TMS) [Doyle 79] を活用する。

本稿では、まず、2章で仮説生成検証過程としての設計の特徴について整理する。それを受けて、3章で本研究で提案する設計支援フレームワークの概要を示し、4章でその実現方法について述べる。5章では設計支援フレームワークのプロトタイプシステムの実装とその上で行った実験的な設計例を紹介する。6章では一連の内容を踏まえて本研究の意義と課題について議論する。最後に、7章で本研究のまとめと展望を論じる。

2. 仮説生成検証過程の構造と設計支援の課題

本章では、仮説生成検証過程に着目した設計支援フレームワークを実現するための前提として、設計プロセスにおける仮説生成検証過程の構造について論じ、それを計算機により取り扱う際の課題を述べる。

2.1 設計における仮説生成検証過程の構造

設計は、設計案を生成してはその検証を行いつつ、その内容を段階的に詳細化していく過程である。つまり、解くべき問題と設計案の両方を仮説として生成しつつ、あわせて、前提となる知識や情報を選んだり、新たな知識を見出したりしながら、その検証を繰り返していく過程で

ある [Schön 82]。また、その際の検証についても、各段階での問題における限られた条件のもとで行われることや、最終的な製品に関しても使用時の条件を完全に把握して行うことは実質的に不可能であることから、検証結果も仮説として位置付けることができる。ひいては、どのような知識や情報を前提にするかという対応関係も仮説となる。つまり、設計で取り扱われる知識や情報はその進行に沿って徐々に増加し具体化していくが、場合によっては以前の時点に戻って設計作業をやり直すこともある。一方で、問題を設定し直したり、複数の代替案を比較検討したりするなどのある種の試行錯誤も不可欠である。このようなプロセスを構造化してとらえる際には、知識や情報が生成されていく過程に着目した時系列の視点と問題と代替案の関係に着目した論理的な視点のそれぞれに立脚することができる。前者のもとで、設計のある時点で利用されている知識や情報の集合を設計状態と呼ぶことにすると、設計の進行に伴って設計状態が推移していくが、同時に、後者のもとでは、着目している問題と採用している案も変化していくことになる。つまり、設計プロセスはそれぞれの視点のもとで独立した構造を持ちつつも、それらは相互に関連していることになる。

上記の内容を説明するために、携帯電話の製品企画において複数の代替案を検討している仮想的な状況を考えてみる。具体的には、図1のような設計状態の変遷と問題・案の変遷を考えてみる。まず、設計状態1では携帯電話の顧客ニーズとして考えられるものを列挙した後、設計状態2ではそれらの中で「デザインが良いこと」というニーズを重要視して設計することを検討している。この案に基づいて設計を進め、コストを検討した結果、その価値に比べてコストが高くなりすぎると判断される(状

態 3) . そこで、デザインの良さを重要度をやや下げ、付加機能の良さをより重視する設計案を検討することとする (状態 4) . その案についてコストを検討すると、価値とのバランスが取れていると判断される (状態 5) . このような過程における問題と案の関係は、状態 4 では状態 2 で設定した問題 2 に戻って再検討している構図となっている . 一方で、設計状態の推移に着目すれば、状態 4 は状態 3 までの設計を引き継いだものであり、その内容は状態 2 に戻ったわけではなく、状態 3 を経た分だけ知識や情報が増えたものになっている . さらに状態が推移し、デザインの良さを重視する案 2.1 が再び採用されることになった場合でも、状態 2 に戻るのではなく、その時点での設計状態の下で案が切り替えられると捉えることができる .

以上のように、設計状態の推移と問題・案の遷移は相互に関連しつつも独立した構造を持っている . このことから、設計支援において、必要に応じて以前の状態に戻ったり、採用している案を柔軟に別のものに切り替えたりすることを可能にするためには、少なくとも、様々な操作や処理に連動させて両視点による構造を把握することが必要となる . これに対して、プロセス管理に着目した設計支援システムのうち、従来からのもの [間瀬 02, 野間口 05, 武内 07, 野中 07] は設計過程の静的な記述を目的としていたために両者のいずれかに着目するに留まっており、設計途中での仮説の動的な切り替えを含めた支援が行えるようにはなっていない .

2.2 問題・案の構造に係わる表現モデルとその課題

上記のうち、問題・案の構造を把握するための情報モデルに関しては、設計根拠 (Design Rationale) [Moran 96] のモデリングとシステム化に関する研究が行われている . 設計根拠とは、人工物の設計の根拠や設計の背後にある知識の総称である . なかでも、問題と代替案の依存関係を扱う枠組みは議論モデルと呼ばれている . 従来から設計根拠を記録するための様々な情報システムが開発されており [Regli 00, Bracewell 04, Burge 08b] , 設計結果の合理的な説明を他者に提供するためだけでなく、設計者自身が自らの意思決定のプロセスを振り返り、理解を深めるためにも役立つとの指摘が行われている [Burge 08a] . その内容は仮説生成検証過程に着目した設計支援フレームワークに求められる情報モデルの一部となるものであると考えられる . しかしながら、一方では、設計根拠は、2.1 節で指摘したところの論理的な視点での設計プロセス表現であり、時系列で進行する設計作業の中で設計者がその作業の論理的な意味を考えながら記述を行うことは余分な負担であると言える [Burge 08a] . また、設計根拠システムは、仮説生成検証に焦点を当てて構築されていることから、設計対象の内容そのものの表現能力が不十分であることも本格的な設計支援システムとしては支障になると考えられる .

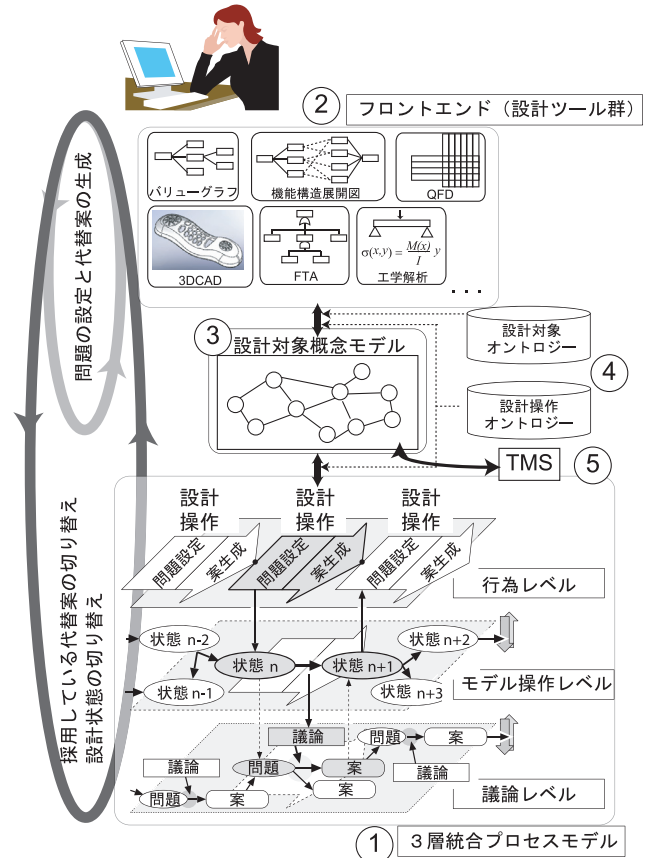


図2 設計支援フレームワーク DRIFT

2.3 設計支援フレームワークの要件

以上の議論により、目指すべき設計支援フレームワークの要件として、以下の各点を挙げるができる .

- 要件 1 設計プロセスにおける時系列的な構造および論理的な構造を、設計者が行う素朴な設計作業と同時に設計根拠モデルの上で記録できること .
- 要件 2 設計作業の記録にあわせて設計対象の内容を表現したり操作したりする形式を、設計者に負担をかけないかたちで用意すること .
- 要件 3 要件 2 のしくみを様々な設計処理に対して柔軟に拡張し、それらを統合できること .
- 要件 4 以上により記録した内容に対して、動的に設計状態や採用している案を切り替えた際に、設計対象の情報を柔軟かつ効率的に変更できること .

3. 設計支援フレームワークの構成

本章では、2 章で導出した要件を踏まえた設計支援フレームワークの基本的な構成を提案する .

3.1 行為・モデル操作・議論の 3 層統合プロセスモデル

本研究では、2.3 節の要件に対して、図 2 に示す設計支援フレームワーク DRIFT (Design Rationale Integration Framework of Three layers) を提案する . このフレーム

ワークの中核は図 2-①の行為・モデル操作・議論の 3 層統合プロセスモデルである。これは 2.3 節の要件 1, つまり、設計状態の変遷と問題・案の構造を含む設計プロセスを時系列で進行する設計作業の副産物として記録するための情報モデルである。

3 層統合プロセスモデルでは、まず、何らかの問題に対して代替案を設定する操作の単位として設計操作を定義する。設計プロセスは問題の設定と案の生成との繰り返しであることから、そのような設計操作の履歴を記録すれば、設計プロセスが表現できるようになる。それと同時に、問題と代替案の対応関係を明示的に把握できるようになり、これを設計根拠モデルを用いて記録する。なお、設計根拠モデルとしては gIBIS [Conklin 88] を用いる。さらに、設計操作を実行する前後での設計状態を記録することにより、設計状態の推移も記録できるようにする。ここで、設計操作を実行する階層を行為レベル、設計状態の推移を管理する階層をモデル操作レベル、問題・案の構造を議論レベルと呼ぶ。DRIFT では、これらを相互に対応付けることによって、素朴な設計作業の実施と同時に設計プロセスの構造が獲得できるようにする。

3.2 設計操作の形式化

設計操作の定義は、DRIFT の起点となるものであると同時に、要件 2 における設計作業の中で設計対象を表現したり操作したりするための枠組みともなることから、特に重要である。しかしながら、設計プロセスにおける表現や操作を汎用的に計算機上で取り扱うことは一般には容易ではないとされている。

上記の課題に関して、中小路は、悪構造問題を支援するための表現系、操作系について、曖昧さを表現できること、部分と全体を同時に概観できること、解と問題とを同時に表現できること、およびこれらを直感的に操作できることが重要であると指摘している [中小路 04, 中小路 06]。これに対して、設計工学の分野では、計算機上での処理を前提としたものではないが、従来から、そのような性質を備える枠組みとして DFX (Design For X) 手法 [Gatenby 90] とも総称される設計方法論が提案されている。DFX は、曖昧な設計の概念を明示的に表現し、ある視点 (X) からの何らかの評価を行う手法の総称である。これらは、曖昧さを含む設計問題や代替案の表現方法およびそれらに対する操作の典型的なパターンを何らかの形で定めたものになっている。例えば、DFX 手法の代表的なものである品質機能展開 (QFD; Quality Function Deployment) [水野 78, Clausing 94] は、設計対象である製品をシステムとして階層的に捉え、そのコストや価値はシステムの構成要素に排他的に分割できること、またシステムは異なる視点で見ることができ、各視点間の項目の間の関連を把握できることを前提として、顧客ニーズや機能、構造などの各視点での項目を検討するための手法となっている。

本研究では、上記の DFX 手法にみられる形式性が一般にも期待できるものとして、各設計手法の内部に潜んでいる表現と操作の形式を設計操作として抽出し定義することによって、図 2-②のように、各手法をツール化して 3 層統合プロセスモデルのフロントエンドとして導入する。すなわち、各種のツールを使い分けながら、ツール上で設計問題を記述したり代替案を記述したり、その評価を行ったりすることを本フレームワーク上での基本操作として位置付ける。これにより、設計者がツール上で行う設計作業と同時に、それらの履歴を DRIFT 上で獲得し管理できるようにする。

3.3 設計対象表現を通じた各種設計手法の統合

設計手法のツール化とそのフロントエンド化によって様々な設計操作を DRIFT に組み込むことができるが、個々の設計手法が扱う内容は限定的である。様々な視点が関連する設計内容を包括的に扱えるようにするためには、要件 3 で指摘したように、それらを統合する必要がある。これに関しては、図 2-③のように、複数のツールが取り扱う対象表現を設計対象概念のネットワークモデルとして実装し、図 2-④のように、設計内容を表現する概念と設計操作をオントロジー [Gruber 93, 溝口 97] のもとで統合する。オントロジーとは、概念化の明示的な規約 [Gruber 93] であり、人工システムを構築する際のビルディングブロックとして用いられる基本概念 / 語彙の体系 (理論) [溝口 97] であり、各種システムの統合を行うための基盤となるものである。本研究では、設計対象の概念モデルを表現するための基本的なオントロジーを用意しておき、それと個々の設計ツールにおける操作内容を表現するオントロジーとの対応関係を定めておくことにより、設計手法間での統合を実現する。これにより、例えば、あるツールにおける記述内容の変更が他のツールの関連する部分の記述内容にも反映されるようにする。

3.4 動的な切り替えのための真理性管理機構の導入

一連のしくみのもとですべての設計状態を複製して管理する場合、仮説や案の動的な切り替えを効率的に行うことが課題となる。すなわち、要件 4 への対処が必要となる。これについては、DRIFT の内部で、図 2-⑤のように、真理性管理機構 (TMS) [Doyle 79] を利用して、1 つの設計対象概念モデルの上で複数の設計状態を管理しておくようにする。TMS は、推論システムの推論結果の履歴の効率的な管理のための手段として開発され、管理する対象となるデータがアクティブか否かを保持した上で、新しいデータが追加されるたびにその更新を行うアルゴリズムである。DRIFT では、TMS により、設計状態をその時点で設計対象概念モデルにおいてアクティブである概念ノードの集合として表現し、3 層統合プロセスモデルにおける設計状態や問題・案と、それらに対応する

概念ノードを対応付けておく．これによって，設計状態の切り替えと案の切り替えをそれらと対応する概念ノードのアクティブ，非アクティブの切り替え操作として実現する．

4. 設計支援フレームワークの実現

本章では，3 章で述べた DRIFT の各構成要素の具体的な実現方法について述べる．

4.1 設計対象概念の表現形式

§1 オントロジーによる表現

設計対象概念モデルについては，3.3 節で述べたように，あらかじめ用意したオントロジーによる様々な概念のクラスのもとに，個々の内容についてのインスタンスを生成する一方，それらの間の関連を表すリンクを付けることにより，設計対象の表現を構成する．なお，以降の説明において，概念のインスタンスを概念ノードと呼ぶ．

まず，上記に向けて，基本となる概念を普遍的に定義するために，対象表現について以下の前提を設ける．

- 設計対象である製品はシステムとして要素の集合として表現できる
- そのような要素への分解は階層的である
- システムを構成する要素はその属性によって特徴付けられる
- そのような表現は対象を観る視点に依存して様々に構成される
- 異なる視点のもとでの階層的なシステム表現の間で要素間の対応関係を記述する必要がある

これらのもとで，オントロジーを体系的に整理するために，以下の 6 つの基本概念を導入する．

製品： 設計対象となる製品の名前．後出の要素ノードを集約する起点となる．

要素： ある視点のもとでの製品情報の構成要素．

階層： 異なる要素の間に階層関係があることを表す N 項関連クラス．

関係： 2 つ以上の要素間に何らかの関係があることを表す N 項関連クラス．

属性： 要素や関係などの設計概念を特徴付ける概念．属性を表す単位をプロパティとして持つ．

属性値： 属性の値を表す概念．具体的な値とスカラ，ベクトルなどのその値のタイプをプロパティとして持つ．

これらのうち，階層と関係については，設計においてはそれらが一つの仮説として設定され得るので，それぞれを概念ノードとして明示的に表現することにする．

以上のもと，フレームワークに統合する設計ツールに依存する個別的概念は上記の基本概念のクラスのサブクラスとして定義する．さらに，これらのクラスのインスタンスである概念ノードには，各ノードを識別するた

めに重複なく割り振られる“ID 番号”と概念ノードの具体的な内容を示す“名前”，および TMS においてある時点の設計状態で“アクティブ”であるかどうかを表現するための真偽値をプロパティとして持たせる．なお，ノードの重複を避けるため，ある概念ノードに同種の関連により関係付けられた概念ノードとして同じ名前プロパティ*1を持つものは生成しないようにする．

§2 概念設計支援手法のもとでの例示

さらに，上記の基本概念に対して，個別の設計ツールで表現される概念を定義しておく．本論文では，提案するフレームワークの概念実証を行うためのテストケースとして，3.2 節で例示した QFD を中心とする概念設計支援手法 [藤田 01] を設計ツールとして位置付けた場合のオントロジー [Nomaguchi 07] を設計ツールで取り扱われる情報，およびその上で行われる操作の形式に基づいて図 3 のように構成する．本例においては，基本概念と合わせて 17 個の概念が定義されている．なお，図は UML (Unified Modeling Language) 形式で記述しており，後出の図でも同記法を用いる．すなわち，QFD などでは，一般に，設計対象を顧客ニーズ，機能，実体など各視点における要素に分解して捉えて，価値やコストなどの属性をそれらの要素に配分することが前提であることを受けて，上記の基本概念を図 3 下部のように展開する．

携帯電話の設計を考えた場合の概念ネットワークモデルは，図 3 のオントロジーのもとで図 4 のように構成できる．例えば，顧客ニーズには“通話がクリア”，“デザインが良い”，機能には，“通信をする”，“画像を記述する”，実体には“LCD モニタ”，“キーボード”などの部品やコンポーネントを挙げる．階層については，例えば，実体“携帯電話”が“LCD モニタ”，“キーボード”などの要素で構成されている場合には，図 4-①のような階層ノードを用いて表す．機能“写真を撮影する”と“バーコードを読み取る”は，いずれもカメラレンズによって実現される機能であるが，これは図 4-②のように機能-実体関係ノードを用いて表す．属性値について，例えば，カメラレンズの相対コストが 20%，相対価値が 5% である場合には，図 4-③のように属性ノードと属性値ノードを用いて表す．

4.2 設計操作の表現と設計対象概念への対応付け

設計ツール上での設計作業は何らかの問題に対して代替案を設定する操作として捉えることができる．それを設計対象概念モデルの上で実施する場合には，設計対象概念モデルに対して，問題を表現している概念ノードを参照しながら，代替案を表現する概念ノードを新たに付け加える操作に対応付けることができる．設計操作はこのことを踏まえて下記のプロパティを持つものとして形式化する．

*1 属性値の場合は値プロパティ

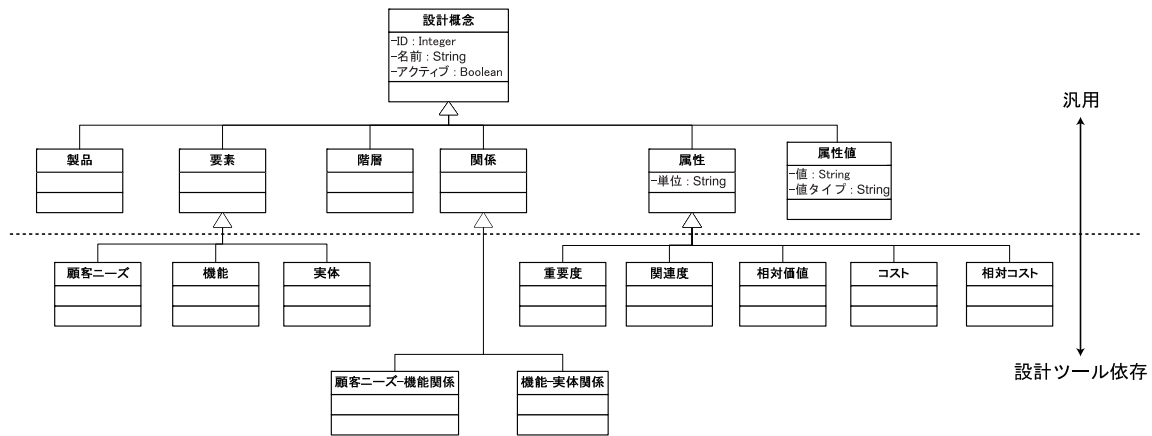


図3 設計対象オントロジーの例

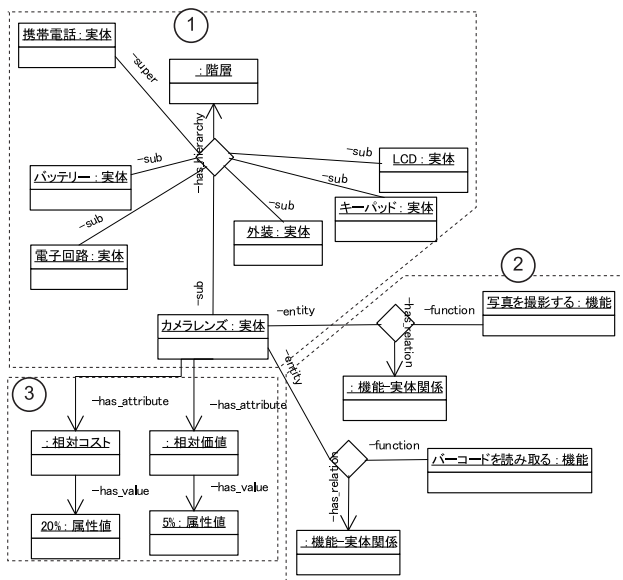


図4 設計対象の概念ネットワークモデルの例

ID: 設計操作を識別するために重複なく割り振られる識別番号。

操作タイプ名: 設計操作を特徴付ける操作の名前。

参照概念ノードリスト: 問題を表現し、設計操作の実行の前提となる概念ノードのリスト。

追加概念ノードリスト: 問題に対する案を表現し、設計操作の実行によって生成、追加される概念ノード。

例えば、“実体の詳細化”を行う操作は「ある実体の下位レベルの構成要素は何か」という問題に対して、何らかのある案を設定する作業である。このような操作は、設計対象概念モデル上では、1つの実体ノードを参照して、その下位レベルの階層を構成する複数の実体ノードを生成し、階層ノードを付け加える操作として実現されることから、参照概念ノードには詳細化の対象となる上位レベルの実体ノードが、追加概念ノードには下位レベルの実体ノードと階層ノードが対応することになる。

4.3 3層統合プロセスモデルのレベル間の連携

設計操作の履歴は3層統合プロセスモデルを通じて設計状態の変遷と問題・案の構造として記録する。これを行うための3層統合プロセスモデルの具体的な内容を図5に示す。なお、図5では、モデルを構成するクラス間の関連を把握しやすくするため、設計状態の変遷と問題・案の構造に対応する全体の構成(a)と、その中で設計操作に関連する部分(b)とに分けて示している。以下の各項では図5の各部分について説明する。

§1 設計状態の変遷管理のための表現の枠組み

4.1節の方法により記述される設計対象概念モデルの内容は、ある時点での設計状態を示すものであり、設計の進行に伴って推移していく。設計状態の変遷を記録した上で、いつでも任意の設計状態に戻ることを可能とするために、設計状態をアクティブな概念ノードのリストのスナップショットとして表現する一方、設計状態の変遷を設計状態をノードとする有向ネットワークグラフにより表現する。これを実現するために、設計状態は下記のプロパティを定義することにより図5-①のように形式化する。

ID: 設計状態を識別するために重複なく割り振られる識別番号。

後続の設計状態: この設計状態から変遷して生成された設計状態。一般に複数のものが存在し、図5-①では `has_following_states` 関連で示される。
アクティブ概念ノードリスト: 設計状態に対応する時点でのアクティブな概念ノードのリスト。図5-①では `has_active_concepts` 関連で示される。

§2 設計状態の変遷管理の設計操作との対応

前項のようにして表現される設計状態の変遷を設計操作の履歴に対応させるために、図5-③に示すように、設計操作と `has_following_state` 関連とを対応させる。つまり、設計操作が1つ実行されると、その設計操作によって追加された概念ノードを含む設計状態が生成され、設計操作実行の直前の設計状態との間に

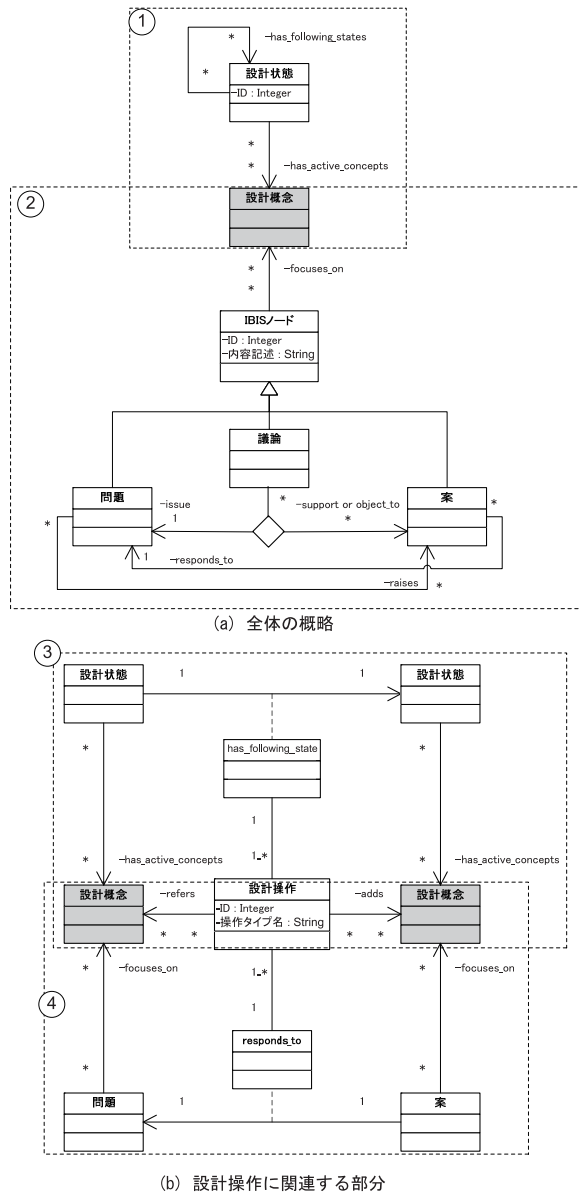


図5 3層統合プロセスモデルのUMLクラス図

has_following_state 関連のリンクが1つ付加されるようにする。ただし、同じアクティブ概念ノードリストを持つ設計状態ノードが既に存在する場合は、新規の設計状態ノードは生成せず、その既存の設計状態ノードへ has_following_state 関連のリンクを付加する。これは、同じ内容を持つ設計状態ノードを複数生成しないようにするための処置である。また、リンクの両端のノードが特定されれば、リンクは一意に定まることとする。このため、1つの has_following_state 関連のリンクには、1つ以上の設計操作が対応する。

§3 問題・案の関係管理のための表現の枠組み

問題と代替案の関係を表現するために、本研究では、代表的な議論モデルである gIBIS[Conklin 88] に基づく表現形式を導入する。このモデルはハイパーテキストによって議論構造を表現するものであり、図5-②に示すよ

うに、問題 (issue)、案 (position)、議論 (argument) の3種類のテキストノード (IBIS ノード) で構成される。議論ノードは、1つの問題に対して提案された複数の代替案のそれぞれがある観点から支持される (support)、あるいは批判される (object_to) ことを表現するノードである。問題ノードと案ノードの間には、案が問題に対応するものであること (responds_to)、問題が案から派生したこと (raises) を示す関連が付けられる。一方、各 IBIS ノードにはその内容に関わる概念ノードを関連付けることにより、設計状態を戻すことなく、採用する代替案を切り替えることが可能にする。これらを実現するために、各 IBIS ノードには以下に示すプロパティを持たせて形式化する。

ID: IBIS ノードを識別するために重複なく割り振られる識別番号。

内容記述: IBIS ノードの内容を説明するテキスト情報。

議論対象概念ノード: IBIS ノードに対応する複数の概念ノード。図5-②では focuses_on 関連リンクで示している。なお、議論対象概念ノードリストのすべてのノードがアクティブである IBIS ノードをアクティブな IBIS ノードと呼ぶ。これはその設計状態において設計者が採用している問題および案を表す。

§4 問題・案の関係管理の設計操作との対応

前項のように表現される問題・案の関係を設計操作の履歴に対応させるために、図5-④に示すように、responds_to 関連と設計操作とを対応させる。設計操作が1つ実行されると、その設計操作の参照概念ノードを議論対象概念ノードとする問題ノード、追加概念ノードを議論対象概念ノードとする案ノードをそれぞれ生成し、両者の間に responds_to 関連のリンクを1つ付加する。ただし、問題ノード、案ノードを生成する際に、対応する設計操作のタイプ名と、議論対象概念ノードリストの内容が同じ問題ノード/案ノードが既に存在している場合は、新規のノードを生成せず、既存の問題ノード/案ノードに対して responds_to 関連のリンクを付加するようにする。これにより、同じ内容を表現する問題ノード/案ノードを複数生成しないようにする。また、リンクの両端のノードが特定されれば、リンクは一意に定まることとする。このため、1つの responds_to 関連のリンクには、1つ以上の設計操作が対応する^{*2}。

また、raises 関連のリンクも、設計操作の実行と同時に下記の手順に従って付加する。

- (1) 実行された設計操作に対応する responds_to リンクにおいて、その問題ノードを I とする。
- (2) I の議論対象概念ノードリストに含まれる概念ノードのうち、少なくとも1つを議論対象概念ノードリストに含む既存の案ノード P を探索する。ただし、

*2 ただし、いずれの設計操作も ID が異なるだけで、操作タイプ名、参照概念ノード、追加概念ノードは同じ内容となっている。

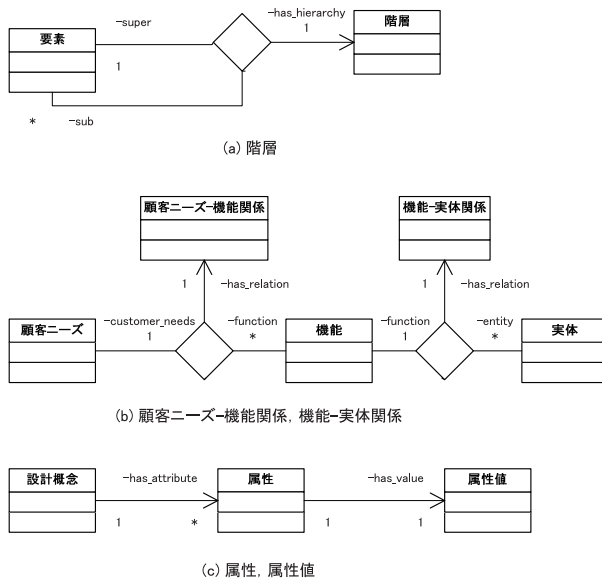


図7 概念間の多重度の定義

ること、属性値については、図7-(c)により、ある属性ノードの属性値ノードはたかだか1つであることを定めておく。それらの規則のもとで複数の概念ノードを付けようとする場合は、最も新しく生成したものを1つ残して、それ以外のすべてを偽とする。

4.6 仮説の切り替えに連動した設計状態の更新

3層統合プロセスモデルによって獲得された設計プロセスの情報をもとに、仮説の切り替えに対応して、設計対象概念モデルの内容を更新することができる。具体的な切り替え操作には、以下の各項に示す設計状態の変更と採用している案の変更の2つがある。

§1 設計状態を変更する場合

設計者が任意の設計状態を選択した後、そのアクティブ概念ノードリストに含まれるノードのアクティブプロパティを真、それ以外のノードを偽とする。

§2 案を変更する場合

設計者が任意の案を選択した後、その議論対象ノードリストに含まれるノードのアクティブプロパティを真とし、4.5節の各項の内容に従ってその他の概念ノードのアクティブプロパティを更新する。この操作は、選択した案の *responds_to* リンク (これは1つしかない) に対応している設計操作を再実行するのと同じことである。

5. 設計支援フレームワークのプロトタイプピン

5.1 実装

提案したフレームワークの概念実証を行うために、DRIFTの実験システムをオブジェクト指向言語 Java により構築した。本実験システムでは、一例として、前述の QFD に基づくコスト価値分析手法 [藤田 01] に基づいた支援シ

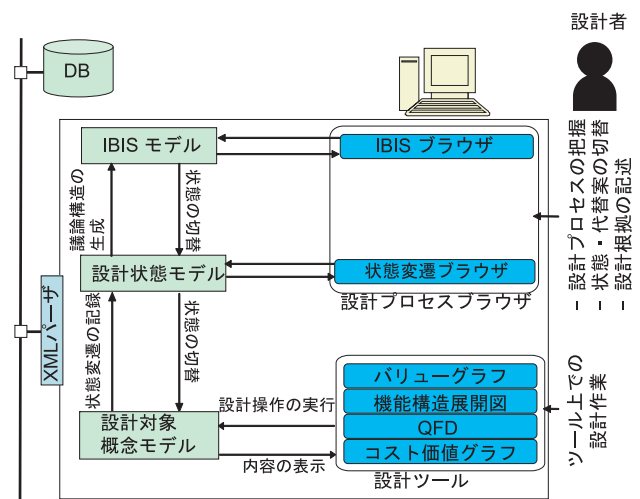


図8 プロトタイプシステムの構成

ステムの構築を目指し、バリューグラフ、機能構造展開図、QFD、コスト価値グラフの4つの設計手法のツールを DRIFT に統合する。統合にあたり、これらの手法の背後にある概念のオントロジーおよび設計操作のオントロジーを定義しているが、前述の図3、図6、図7はそれらの一部である。

図8にこのプロトタイプシステムのシステム構成を、図9に実行画面を示す。システム画面は、図9右側の設計ツール部と、図9左側の設計プロセスブラウザ部の2つの部分から構成されている。設計ツール部では、DRIFTに統合された複数のツールをタブによって切り替えて使い分けができるようになっている。図9ではQFD二元表ツールが表示されている。DRIFT上での基本操作は設計ツールを用いて仮説的な代替案の設定とその検証を行う作業であり、設計者は設計作業の内容に応じて適切なツールを用いてこれらの作業を進める。設計ツール上の操作はシステムによって自動的に記録され、あらかじめ定義された設計操作テンプレートによって設計状態の推移と問題・案の構造が生成される。これらは設計プロセスブラウザ上に表示される。設計者は設計プロセスブラウザ上で、他の代替案に切り替えたり、以前の状態に戻って問題を再検討したりできる。また、必要に応じて、ある時点において代替案を選択した根拠を、議論ノードを用いて記述することができる。

なお、設計対象概念モデルと設計プロセスの情報はXML (eXtensible Markup Language) 形式で保存したり読み込んだりできるようにしている。

5.2 設計例題の構成

上記のような実験システムの適用例題として、携帯電話の概念設計をQFDに基づくコスト価値分析手法のもとで行う場合を考える。近年の携帯電話市場では、多様な顧客ニーズに合わせて様々な付加機能を持つものが開

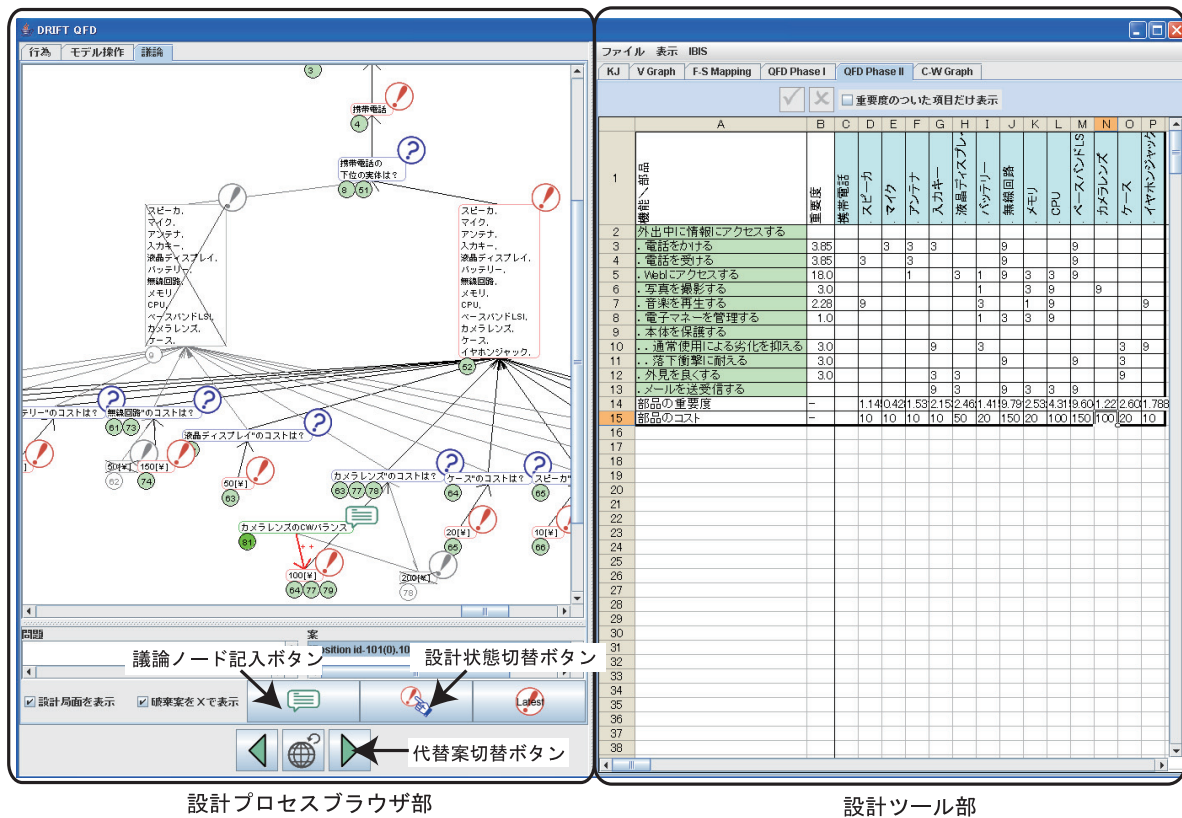


図9 プロトタイプシステムの実行画面

発されている一方で、コスト削減競争も厳しい。このため、設計者は、対象とする製品の顧客ニーズを明確にしつつ、各部品について顧客ニーズからみた重要度に応じてコストを配分するように製品の開発コンセプトを定める必要がある。それに対して、QFDに基づくコスト価値分析手法はそのような設計を合理的に行うための有効な方法の一つである。

次節以降に示す適用例では、設計者が、実験システムに統合された複数の設計支援ツールを使いながら、様々な問題に対して仮説的に代替案を設定しながら設計を進めていく状況を考える。その際に問題を検討していく手順 [藤田 01] をおおむね以下のように想定する。

- (1) バリュースコアを用いて、製品の顧客ニーズについての検討を行う。
- (2) 機能構造展開図を用いて、製品の機能と実体についての検討を行う。
- (3) QFD 二元表を用いて、(1) による顧客ニーズと (2) による機能と実体の間の関係について検討した後、顧客ニーズの重要度を設定して、それを実体の重要度へと展開する。また、展開した重要度の相対値(相対価値)を計算する。
- (4) 実体のコスト目標を検討する。その相対値(相対コスト)を計算する。
- (5) コスト価値グラフを用いて、(3) による各実体の相対価値と(4) による相対コストのバランスを見て、

各案の妥当性を検証する。このとき、バランスの取れている実体は、右上がり 45° の線(相対価値と相対コストの比が 1 であることを意味する)に近い部分にプロットされる。

- (6) バランスの取れていない実体があれば、前の段階に戻って案の変更を検討する。

このように、本手法では、仮説として設定された設計案をコスト価値バランスの観点から検証し、種々の案を検討しながら、優れた設計案を見つけていく構図となっている。ただし、上記はあくまでも標準的な手順であって、実際の状況では様々な手順を交錯させながら種々の検討を行う必要がある。例えば、QFD 二元表を作成する段階で顧客ニーズや機能を検討し直すことなども一般に起こりえる。DRIFT および実験システムでは、そのような場合においても、設計プロセスにおけるすべての操作を記録し、設計状態の変遷とともに問題と代替案の依存関係を管理することができるようになっている。

以下の各節では、2.3 節で述べた 4 つの要件に関連させながら、DRIFT の機能をそれぞれに説明する。

5.3 設計ツールによる設計対象の表現と操作

図 10 は、DRIFT に統合されたバリュースコアツールを利用して、顧客ニーズ項目を検討している様子である。図 10-① では、“良い携帯電話通話” というトップ項目の下に、通話がクリア、デザインが良いなど 4 つの項目を

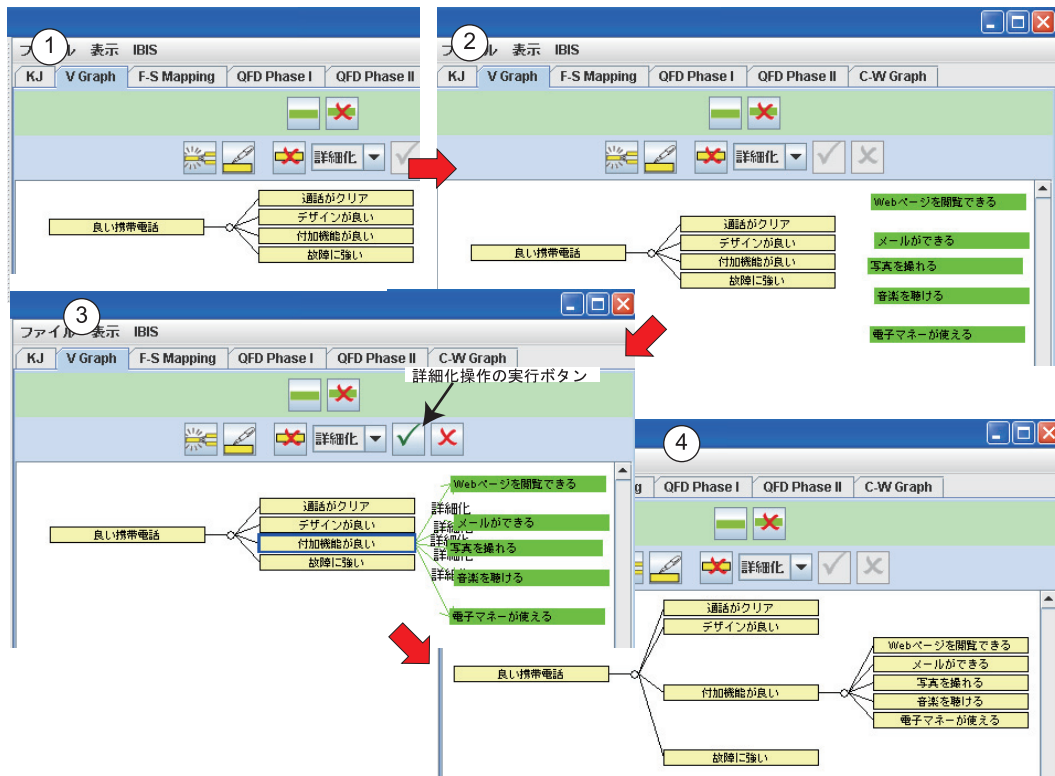


図 10 バリュグラフによる顧客ニーズ詳細化の操作

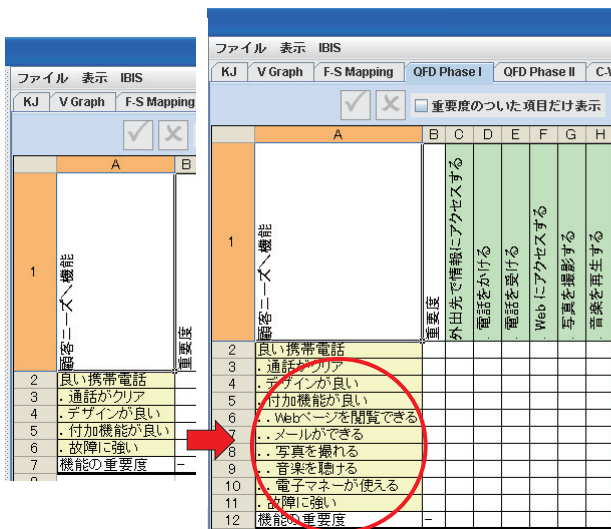


図 11 バリュグラフ上の操作の QFD 二元表への反映

列挙している。ここで、さらに詳細な項目として、Web ページを閲覧できる、メールができる、写真を撮れる、音楽を聴ける、電子マネーが使える、の 5 項目を列挙し (図 10-②)、いずれも“付加機能が良い”という項目の下位の顧客ニーズと位置付け、ノード間にリンクを付けている (図 10-③)。リンクをつけた後で、設計操作実行ボタンをクリックすると、顧客ニーズ詳細化の設計操作が呼び出され、詳細化が実行される (図 10-④)。なお、この操作の結果は他の設計ツールにも反映される。例えば、QFD 二元表では、図 11 のように顧客ニーズ項目が更新される。

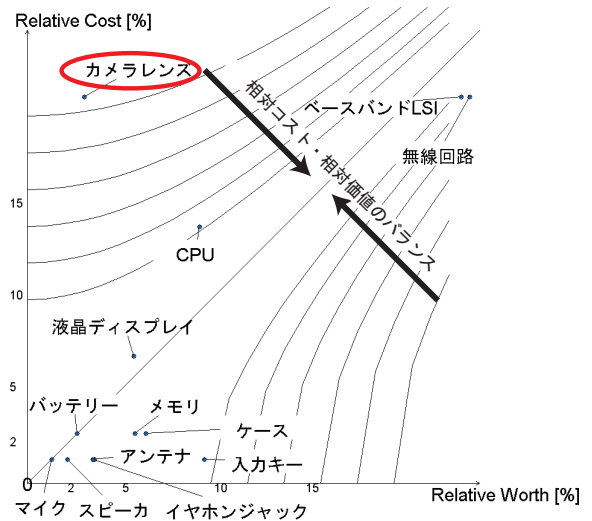


図 12 当初の設計案の下でのコスト価値グラフによる評価

続いて、図 9 右側に示すような QFD 二元表を作成して各実体の重要度を算出し、その上で各実体のコスト目標を設定した後に、コスト価値グラフによって相対価値と相対コストのバランスを見て、現時点で採用している代替案の検証を行う。当初案でのコスト価値バランスは図 12 のようになり、カメラレンズのコストがその価値に比べて高いことが判明する。そこで、このバランスを調整するための設計変更を行う。具体的には、カメラレンズの (A) 相対コストを下げるか、(B) 相対価値を上げるかのいずれかを行うことになる。図 13 は、(A) の方針に従って、現状のカメラレンズコスト目標である 150 円

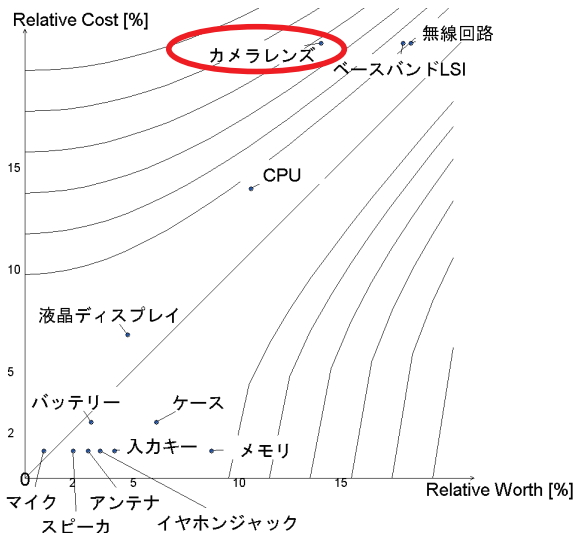


図 15 再検討案の下でのコスト価値グラフによる評価

とができるが、この例では、図 17(b) に示すように、設計状態 54 番の時点に対して“カメラレンズのコスト価値バランス”の観点からコスト目標案 100 円が採用されていることを記述している。

5.5 仮説の切り替えに連動した設計対象情報の動的変更

IBIS 形式で獲得された設計プロセスに対して、設計者が代替案の切り替え操作を行うことにより、切り替えた代替案に関連する内容を設計支援ツール上に呼び出し、その下で検討し直すことができる。

例えば、設計状態 54 番での設計変更の後、100 円のコスト目標は技術的に難しいことが判明したとする。設計プロセスブラウザには、図 17 に示すように、現在採用している案とともに、以前の案である 150 円のコスト目標が記録されている。これを再び採用し、他の設計変更を改めて検討し直すことを考える。この場合には、設計根拠ブラウザ上で、破棄された案を示す IBIS ノードを選択し、「代替案の切替」ボタンをクリックすることにより、IBIS ノードに関連づけられている設計操作が実行される。これにより、図 18 に示すように、破棄されていた代替案が再びアクティブになり、QFD 二元表やコスト価値グラフの内容が更新されることになる。

上記の「代替案の切替」の操作では、以前の設計状態に戻るのではなく、現在の状態において改めて設計操作を実行している。仮に、前者に相当する以前の設計状態に完全に戻りたい場合には「設計状態の切替」ボタンをクリックすることで、それを行うことができる。そうではなく、後者に相当する本例の場合は、図 18 に示すように、新たに 56 番の設計状態が、150 円のコスト目標案を設定した状態として新たに追加されている。この状態では、設計状態 52~55 番の設計操作により記述した内容が現在の状態に引き継がれていることになる。ここでは、設計者が改めてその案を採用した根拠を“コスト削減は

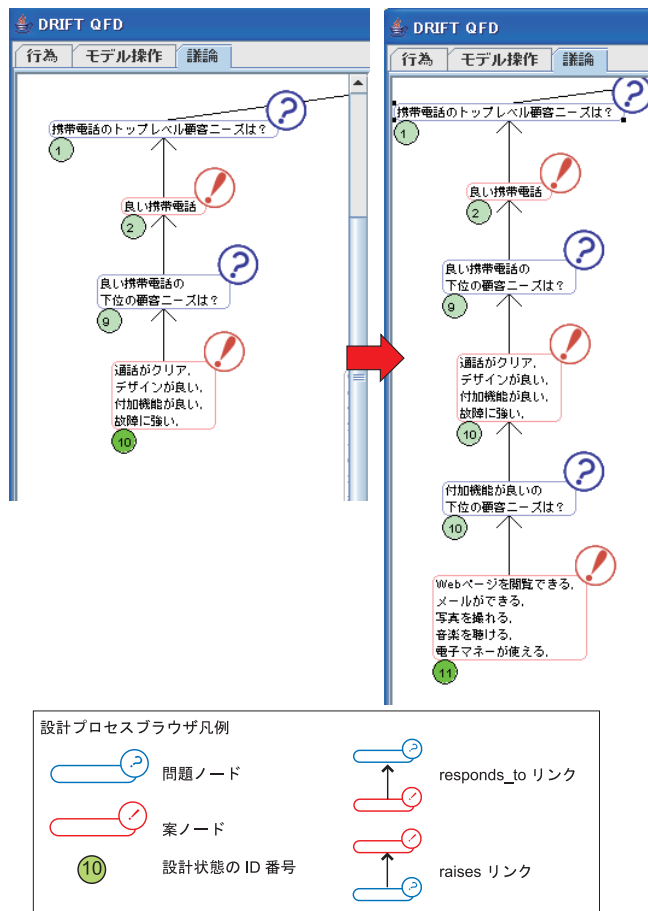


図 16 顧客ニーズ詳細化のプロセスの記録

技術的に困難”として議論ノードに記しているが、前節で追加した議論ノードも残っている。つまり、設計の各状態でどのような判断を行ってどのような代替案を検討したのかがすべて記録されることになる。

6. 本研究の意義と課題

5章で示した設計例は、DRIFTに基づいて構築された設計支援システムの上で設計を行うことにより、設計プロセスにおける設計状態の変遷、仮説として検討された問題と代替案の構造を動的に管理できることを示している。これにより、本研究で提案した DRIFT のフレームワークは、2.3 節で示した 4 つの要件を実現していることが実証されたと考えられる。具体的には、まず 5.3 節により要件 2、すなわち設計作業の記録にあわせて設計対象の内容を表現したり操作したりする形式を、各種設計ツールをフロントエンドとすることで設計者に負担をかけないかたちで用意できたことが示されている。また、要件 3、すなわち複数の設計ツールがオントロジーのもとで統合されていることが示されている。次に、5.4 節により要件 1、すなわち設計プロセスにおける時系列的な構造および論理的な構造を、設計ツール上で設計者が行う作業と同時に gIBIS 形式に従って記録でき、代替案の

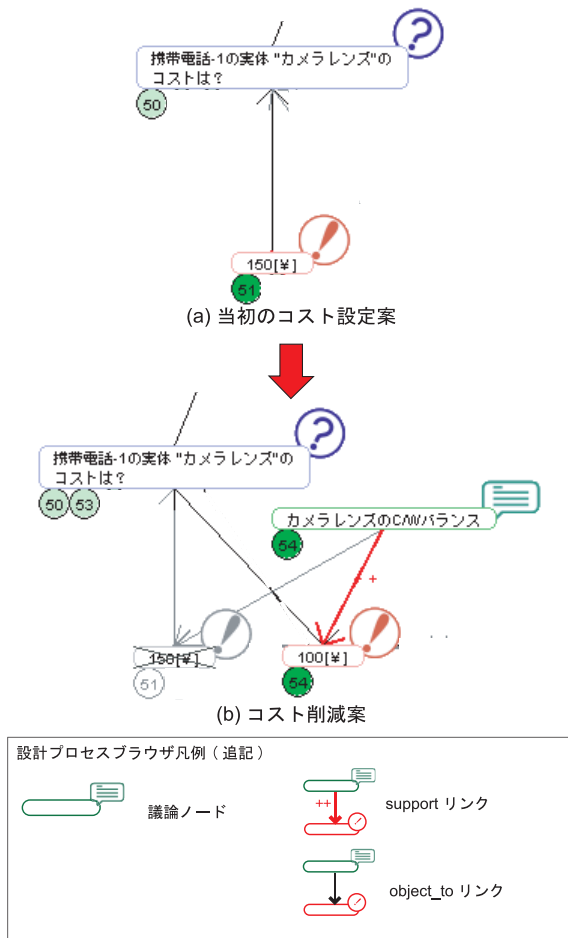


図 17 コスト設定に関する仮説の切り替えと根拠の記録

採用の根拠を自由に記述することができることが示されている。最後に 5.5 節により要件 4, すなわち動的に設計状態や採用している案を切り替えた際に、設計対象の情報を柔軟かつ効率的に変更できることが示されている。

2.2 節で述べているように、本研究は設計プロセスを記録するための枠組みに焦点を当てている点で、設計根拠に関する従来研究と関連している。一方で設計根拠記録の枠組みは、一般に他者への設計根拠の提供に焦点を置いており、また設計対象の内容そのものの表現能力が不十分であるため、本格的な設計支援システムとしての利用に問題が残っている。これに対して、そもそも本研究では設計支援のフレームワークの構築を目的としており、設計プロセスを記録した上で、代替案を切り替えたり、以前の設計状態に戻ったりする設計に不可欠な作業を容易に行うための機能の提供に焦点を置いている点異なる。また、本研究で提案するフレームワークでは、フロントエンドとして複数の設計ツールを統合しておき、設計者がそれらを利用して作業を行えるようにしている。設計プロセスの記録は、これら設計者にとって馴染みのある設計ツール上の作業の副産物として行われる。このアプローチは従来の設計根拠研究にはなかったものである。設計根拠研究では、設計根拠記録の負担の問題に加え

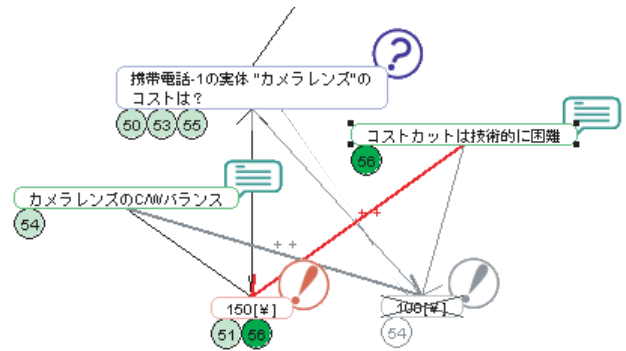


図 18 以前のコスト設定案の再検討

て、獲得した設計根拠の信頼性や再利用性、およびそれらを担保するための設計根拠を記録する設計者に対するインセンティブなどが問題として指摘されている [Conklin 96]。これらは、設計作業と設計根拠記録作業の乖離、および設計者と設計根拠利用者の乖離によって発生する問題と考えられる。前者の乖離について、本研究で提案したフレームワークにおいては、設計プロセスの記録は設計ツール上の作業の副産物として行われるため、直接的な問題とはならない。後者の乖離については、本研究で提案したフレームワークで獲得した設計プロセスをいわゆる設計根拠として他者に提供する場合には、記録された設計プロセスの中から適切な内容を抽出したり、場合によっては必要な情報を補完することも必要となると考えられる。この点については今後の研究課題である。

一方、本フレームワークのもとでの設計支援の様々な可能性は設計操作のテンプレートに埋め込まれたモデル操作がその実現上の鍵となっているが、それらの内容は事前に抽出された設計対象のオントロジーに基づいて記述されている。つまり、5.3 節で示した、複数の設計ツールによって表現される内容の一元的管理についての機能は、設計対象の概念のオントロジーに基づいて行われ、また、5.4 節および 5.5 節で示した、設計ツール上の設計のプロセスの獲得、管理についての機能は、設計操作のテンプレートに基づいて行われている。言い換えれば、導入したオントロジーによって規定された概念から外れる設計操作は直接的には対応することができない。このため、5 章に示した実験システムで取り上げたよりも複雑な設計問題を支援するためには、背後にあるオントロジーを拡充し、あわせて設計操作のテンプレートを拡充していくことが必要となる。この意味において、設計知識や設計操作における形式を明らかにしていくことは知識管理型設計支援システムをより高度なものとしていく上での基盤的な課題であると言える。

オントロジーの作成は、4.1.2 節および 4.4.1 節でも述べたように、フレームワークに統合する設計ツールで取り扱われる情報、およびその上で行われる操作の形式を抽出することで行う。上記の課題については、3.2 節で

論じた DFX 手法にみられる形式性を、例えば、設計の下流段階で用いられる CAE (Computer-Aided Engineering) や 3 次元形状システムなどにおいても抽出することができるかどうかは鍵となる。一般にあらゆる設計ツールにおいて、何らかの設計概念の表現とそれの上での操作が提供されていることを踏まえれば、これらについても同様の考え方のもとで拡張が可能であると考えられる。これに関して、著者らは、5 章で示した QFD を始めとする DFX 手法に基づく概念設計支援 [Nomaguchi 07]、複数製品の共通化と顧客ニーズに対応した多様化を適切に企画するための製品系列設計支援 [Nomaguchi 06]、工学解析モデリングプロセス支援 [野間口 09] について、それぞれにおける設計操作や概念についての研究を進めており、これらをフレームワークに組み込むことによって、DRIFT の下で動的に管理される設計内容を拡張することができるものと考えている。

なお、DRIFT に基づく設計支援システムの有効性を確認するために、5.1 節で述べたプロトタイプシステムを、主に機械製品の設計を担当する企業の技術者、および大学院の設計演習において機械分野の各種製品の設計開発プロジェクトに取り組む学生に実際に利用してもらい、調査を行っている。それによれば、フレームワークの基本機能については、複数の設計ツールの統合機能により効率的にツールを利用できる、設計における試行錯誤過程が残ることにより次機種開発で役に立つ、記録した設計プロセスをナレッジとして使える、といった肯定的な評価が得られている。詳細な調査、分析、およびフレームワークへのフィードバックは今後の課題である。

また、設計プロセスを仮説の観点から理解することについては、仮説生成の推論形態であるアブダクション [米盛 81] に着目し、その推論過程の形式化に関する研究 [Takeda 90] や、推論過程を計算機で直接支援することを目的とする研究 [下村 06] が行われている。アブダクションにおいては一般に、妥当でないものも含めて複数の仮説が生成される可能性があるため、設計支援に用いる際には仮説の検証の支援が課題となる [下村 06]。本研究は計算機工学的立場に基づき、3 層統合プロセスモデル、設計対象概念モデル、設計ツール群を、4 章で述べたアルゴリズムにより自動的に相互に関連付けて計算機上で統合管理することにより、生成された複数の仮説の動的な管理を行って仮説生成検証を支援するフレームワークを提案するものである。このため上記の立場とは相補的であり、両者を連携させることでより高度な設計支援の実現ができると考えられる。

7. まとめ

本論文では、仮説生成検証の動的な展開に焦点を当てた設計支援フレームワークとして DRIFT を提案した。DRIFT は、様々な設計支援ツールの上の設計プロセスを

一元的に管理するための行為・モデル操作・議論の 3 層統合プロセスモデルを備えており、各種の設計支援ツール上で設計者が行ったすべての設計操作を自動的に獲得すると同時に、その操作によって推移した設計状態や仮説的に設定される複数の問題、代替案の関係を管理しておくことにより、状態の切り替えと同期して設計支援ツールに記述された内容を動的に変更することを可能にする。あらかじめ用意する設計対象表現についてのオントロジーや設計手法のツール化と設計操作のテンプレート化などに依存するものの、DRIFT に基づいた設計支援システムのもとでは、仮説生成検証が効率的に行えるようになるのみならず、通常は暗黙的なものに留まっている設計の進捗や背後にある根拠を明示的に獲得して、それらを共有したり再利用したりすることも期待できる。

謝 辞

本研究の一部は科学研究費補助金 (若手研究 (B) 19760101) の助成を受けて実施したものである。本研究の遂行にあたり、大阪大学大学院工学研究科の大沼充史氏、飯高智史氏、田口智祥氏との議論が大変参考になった。ここに感謝申し上げる。

◇ 参 考 文 献 ◇

- [Bracewell 04] Bracewell, R. H., Ahmed, S., and Wallace, K. M.: DRed and Design Folders, a Way of Capturing, Storing and Passing on, Knowledge Generated during Design Projects, in *Proceedings of DETC'04 ASME 2004 Design Engineering Technical Conf. & Computers and Information in Engineering Conf.*, DETC2004-57165 (2004)
- [Burge 08a] Burge, J. E. and Bracewell, R.: Design Rationale: Researching under Uncertainty, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 22, No. 4, pp. 311-324 (2008)
- [Burge 08b] Burge, J. E. and Bracewell, R.: Special Issue: Design Rationale, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 22, No. 4, pp. 309-310 (2008)
- [Clausing 94] Clausing, D.: *Total Quality Development. A Step-By-Step Guide to World-Class Concurrent Engineering*, ASME Press (1994), (邦訳: 品質・速度両立の製品開発. Total Quality Development, 富士ゼロックス TQD 研究会訳, 日経 BP 社, (1996))
- [Conklin 88] Conklin, J. and Begeman, M. L.: gIBIS: A Hypertext Tool for Exploratory Policy Discussion, *ACM Transactions on Office Information Systems*, Vol. 6, No. 4, pp. 303-331 (1988)
- [Conklin 96] Conklin, E. J. and Burgess-Yakemovic, K. C.: A Process-Oriented Approach to Design Rationale, in Moran, T. P. and Carroll, J. M. eds., *Design Rationale: Concepts, Techniques, and Use*, pp. 393-427, Lawrence Erlbaum Associates (1996)
- [Doyle 79] Doyle, J.: A Truth Maintenance System, *Artificial Intelligence*, Vol. 12, No. 3, pp. 231-272 (1979)
- [藤田 01] 藤田 喜久雄, 西川 武志: 製品の高付加価値化とその品質機能展開による設計評価法, 日本機械学会論文集 (C 編), Vol. 67, No. 656, pp. 1202-1209 (2001)
- [Gatenby 90] Gatenby, D. A. and Foo, G.: Design For X (DFX) - Key to Competitive, Profitable Products, *AT&T Technical Journal*, Vol. 69, No. 3, pp. 2-13 (1990)
- [Gruber 93] Gruber, T. R.: A Translation Approach to Portable Ontologies, *Knowledge Acquisition*, Vol. 5, No. 2, pp. 199-220 (1993)
- [間瀬 02] 間瀬 久雄, 絹川 博之, 森井 洋, 中尾 政之, 畑村 洋太郎: 思考過程の思考展開図表現に基づく機械設計支援システム, 人工知能学会論文誌, Vol. 17, No. 1, pp. 94-103 (2002)

- [水野 78] 水野 滋, 赤尾 洋二: 品質機能展開, 日科技連出版社 (1978)
- [Moran 96] Moran, T. P. and Carroll, J. M.: *Design Rationale – Concepts, Techniques, and Use*, Lawrence Erlbaum Associates, Inc. (1996)
- [中小路 04] 中小路 久美代, 山本 恭裕: 創造的情報創出のためのナレッジインタラクションデザイン, 人工知能学会論文誌, Vol. 19, No. 2, pp. 154–165 (2004)
- [中小路 06] 中小路 久美代, 山本 恭裕: 創発のためのソフトウェア, 鈴木 宏昭 (編), 知性の創発と起源, pp. 111–131, オーム社 (2006)
- [野間口 05] 野間口 大, 下村 芳樹, 富山 哲男: 設計者の思考過程のモデルを利用した設計知識管理システム, 人工知能学会論文誌, Vol. 20, No. 1, pp. 11–24 (2005)
- [Nomaguchi 06] Nomaguchi, Y., Taguchi, T., and Fujita, K.: Knowledge Model for Managing Product Variety and its Reflective Design Process, in *Proceedings of the 2006 ASME Design Engineering Technical Conf. & Computers and Information in Engineering Conf.*, DETC2006-99360 (2006)
- [Nomaguchi 07] Nomaguchi, Y. and Fujita, K.: Ontology Building for Design Knowledge Management Systems Based on Patterns Embedded in Design-for-X Methodologies, in *Proceedings of 16th International Conference on Engineering Design (ICED 07)*, Paper No. 442 (2007)
- [野間口 09] 野間口 大, 田口 智祥, 藤田 喜久雄: 工学解析モデリングのための知識管理フレームワークについての考察, 日本機械学会論文集 C 編, Vol. 75, No. 756, pp. 2181–2190 (2009)
- [野中 07] 野中 紀彦, 清水 勇喜, 用田 敏彦, 横張 孝志, 西垣 一朗: 設計プロセスとナレッジを融合した設計誘導システムの開発, 日本機械学会論文集 C 編, Vol. 74, No. 737, pp. 225–232 (2007)
- [Regli 00] Regli, W. C., Hu, X., Atwood, M., and Sun, W.: A Survey of Design Rationale Systems: Approaches, Representation, Capture and Retrieval, *Engineering with Computers*, Vol. 16, No. 3–4, pp. 206–235 (2000)
- [溝口 97] 溝口 理一郎: オントロジー工学序説 - 内容指向研究の基盤技術と理論の確立を目指して, 人工知能学会誌, Vol. 12, No. 4, pp. 559–569 (1997)
- [Schön 82] Schön, D. A.: *The Reflective Practitioner – How Professionals Think in Action*, Basic Books Inc (1982), (邦訳: 省察的実践とは何か – プロフェッショナルの行為と思考 –, 柳沢 昌一, 三輪 建二 監訳, 鳳書房, (2007))
- [下村 06] 下村 芳樹, 吉岡 真治, 武田 英明, 富山 哲男: アブダクションに基づく設計者支援環境の基本構想, 日本機械学会論文集 C 編, Vol. 72, No. 713, pp. 274–281 (2006)
- [Takeda 90] Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawa, H.: Modeling Design Processes, *AI Magazine*, Vol. 11, No. 4, pp. 37–48 (1990)
- [武内 07] 武内 雅宇, 小路 悠介, 來村 徳信, 林 雄介, 池田 満, 溝口 理一郎: 知識成長過程を指向した設計意図知識管理システムの構築, 人工知能学会論文誌, Vol. 22, No. 3, pp. 263–275 (2007)
- [富山 02] 富山 哲男: 設計の理論, 岩波書店 (2002)
- [米盛 81] 米盛 裕二: パースの記号学, 勤草書房 (1981)

著者紹介



野間口 大(正会員)

大阪大学大学院工学研究科機械工学専攻助教。博士(工学)。2002年東京大学大学院精密機械工学専攻博士課程修了。東京大学人工物工学研究センター研究員,大阪大学大学院工学研究科助手を経て,2007年4月より現職。研究分野は設計工学,設計支援のための知識管理手法とそのシステム化など。精密工学会,日本機械学会,The Design Society,設計工学会,日本計算工学会,認知科学会,情報処理学会各会員。



藤田 喜久雄(正会員)

大阪大学大学院工学研究科機械工学専攻教授。工学博士。1990年大阪大学大学院工学研究科産業機械工学専攻後期課程修了。大阪大学工学部助手,講師,助教授を経て,2002年8月より現職。研究分野は,設計学・設計方法論,設計のための情報システム,設計最適化など。日本機械学会, American Society of Mechanical Engineers (ASME), The Design Society, 計測自動制御学会, システム制御情報学会, 設計工学会各会員。

〔担当委員: 來村 徳信〕

2009年7月1日 受理