知識ベース CAD 活用のための知識コードコンバータ *

—Maxis C^3 (Cubic) の開発—

野間口大 ** 下村芳樹 *** 後藤範恭 † 小池一郎 †

Development of Knowledge-code Converter for Activating Knowledge-based CAD

Yutaka NOMAGUCHI, Yoshiki SHIMOMURA, Yasunori GOTOH and Ichiro KOIKE

This is a report on a new methodology to manage design knowledge by utilizing a knowledge-based CAD and a prototype system named Maxis C^3 (Cubic; CAD knowledge Code Capacitor), which is being developed using our methodology. C^3 facilitates (i) the automatic generation of a knowledge code for a knowledge-based CAD by processing design procedure documents written in a natural language, such as English or Japanese, and (ii) automatic generation of a design procedure document written in a natural language from the knowledge code. The features of the system facilitate document-based design knowledge management which reduces the designer's load to encode and maintain design knowledge, because it is easier for a designer to treat a natural language description than a coded description.

Key words: knowledge-based CAD, design knowledge, design document, knowledge management

1. 序 論

近年、企業の競争力の源泉として知識の重要性が指摘されており、知識管理は企業における重要な課題の 1 つとなっている¹⁾。設計研究の分野においても、設計知識管理は今や主要な研究トピックの 1 つであり²⁾、計算機を用いた設計知識管理の実現を目指したシステムの研究も行われている (例えば³⁾ など)。こうした背景の下、Dassault 社の CATIA V5 や Electronic Data Systems 社の Unigraphics など、商用の知識ベース CAD が相次いでリリースされている。これらの知識ベース CAD はハイエンドの 3 次元幾何処理能力に加えて知識ベースシステムを備えており、設計におけるルールや制約をコード化して知識ベースに蓄積しておくことによって半自動設計を可能とするものである。これら知識ベース CAD の機能による設計のより一層の効率化が期待されている⁴⁾ が、同時にその導入効果に対する疑問も投げかけられている。

著者らは設計知識管理の実現に向けて知識ベース CAD を効果的に活用するために Maxis C^3 (Cubic; CAD knowledge Code Capacitor) と呼ぶシステムを開発した。 C^3 は、知識ベース CAD が処理することが可能である知識コードと自然言語で記述されたテキストとの相互変換を実現するコンバータである。 C^3 により設計者は、自然言語で設計手順書を記述することによって自動的に知識コードを生成することが可能となると同時に、知識ベース CAD 上で実行した操作をもとに自動的に設計手順書を生成することが可能となる。本論文では C^3 の開発の基礎となっている方法論およびその実装について、日本語版 C^3 を例に述べる。

Fig. 1 Information flow of design activities (before introducing knowledge-based CAD)

2. 知識ベース CAD の効果と課題

本節では知識ベース CAD の効果と課題について、知識ベース CAD を実際に導入した自動車部品設計会社の設計部門における事例分析に基づいて考察する。

この企業で知識ベース CAD 導入以前に行われていた設計業務を IDEF0 形式⁵⁾ で表現したものを図 1 に示す。まず設計の上流段階において、部品への要求仕様である設計基準書および意匠的な要求仕様としての意匠面が決定される。これらの要求に対して設計者は、過去に設計された同様の部品の図面を参照するなどして、部品の寸法・形状およびその決定方法を記した設計手順書を作成する。設計手順書は、設計基準書に記述された要求仕様を満たすために必要な設計ルールや設計手順を文書として明示したものである。設計手順書は CAD オペレータに手渡され、CAD オペレータは設計手順書に従って部品の CAD モデルを作成する。

知識ベース CAD を導入することにより、設計業務は図 2 のように変化した。この設計業務において、設計手順書の作成までの過程は知識ベース CAD 導入以前と同一であるが、設計手順書を元に知識コードを作成する過程が新たに加わっている。この知識コードは知識ベース CAD の半自動設計機能を利用するために設計手順書に記述されている設計制約や設計ルール、設計手順を知識ベース CAD の書式に従ってコード化

* 原稿受付 平成 16 年 8 月 12 日

** 正 会 員 大阪大学大学院 (吹田市山田丘 2-1)

*** 正 会 員 東京大学人工物工学研究センター (目黒区駒場 4-6-1)

† (株) マキシス (名古屋市中区東横 1-13-3)

Fig. 2 Information flow of design activities (after introducing knowledge-based CAD)

したものである。この知識コードを用いることにより、CADモデル作成作業を知識ベースCADの知識ベースモデリング機能のサポートの元に行うことができるため、CADオペレータの作業負担が軽減される。

しかしながら同時に、知識ベースCADの導入に伴う問題点も確認された。まず、知識コード作成のためには知識コードのコーディングに関する知識が必要であり、その習得は設計者にとってもCADオペレータにとっても大きな負担となる。次に、知識コードの表現形式は各知識ベースCAD間で一般には互換性がないため、何らかの理由により利用する知識ベースCADを変更する必要がある場合には知識コードを再度コーディングする必要がある。この問題は知識の再利用性の観点から見て、大きな欠点である。

一旦知識がコード化された後も、知識コードのメンテナンスが必要であるという問題が確認された。人工知能の研究分野において、エキスパートシステムを永続的に活用するためには計算機に蓄積された知識の陳腐化を抑止するため、新しい概念の追加や誤った知識、矛盾した知識、冗長な知識の削除を行って知識のメンテナンスを行い続けなければならないことが指摘されており⁶⁾、同様の指摘が知識ベースCADにも当てはまることになる。しかしながら、日本語で記述された設計手順書に対するメンテナンスは実際に行われているものの、知識コードを直接メンテナンスすることは設計者に対してもCADオペレータに対しても新たな大きな負担となる。

以上の考察を裏付けるために、この企業の設計部門における年度毎のCADモデリング(a)、知識コード作成(b)、知識のメンテナンス(c)に費やされる作業量の推移を調査した。その結果を表1に示す。調査の対象としたのは自動車に関連するある部品の設計作業量である。この部品は1年に3度モデルチェンジされるため、そのつど新規の設計が必要となる。なお、単位は人日である。

表1において、知識ベースCAD導入以前と比較して、導入初年度はCADモデリングの作業量が大幅に削減されている(24→2)。しかしながら同時に、知識コード化に費やされる作業量が新たに増えており(50)、作業量の合計は導入以前よりかえって増加していることが分かる(24→62)。知識コード化作業は初年度に一通り終えているため、次年度以降減少している(50→9)。CADモデリング作業量は引き続き抑えられているが、知識コードのメンテナンスに費やされる作業量はそれほど減っていない(10→6)。このため、作業量の合計は知識ベースCAD導入以降と比較してやや減少した程度に留まっている(17)。

以上の議論により、知識ベースCADを実際の設計作業にお

Table 1 Variation of loads by the introduction of knowledge-based CAD

	Load of designer (ManDay)*			
	(a)	(b)	(c)	Sum.
Before				
Knowledge-based CAD	24	-	-	24
1st year after introducing				
Knowledge-based CAD	2	50	10	62
Afterward	2	9	6	17

* (a):CAD modeling, (b):Knowledge encoding, (c):Knowledge maintenance

いて効果的に利用する一つの鍵は、知識コードの作成と、作成した知識コードのメンテナンスの双方の効率化を実現することであると考えられる。また利用する知識ベースCADを変更する場合に備えて、作成する知識コードは知識ベースCADに依存しない形態で管理されることが望ましい。本研究ではそのために、日本語で記述された設計手順書の形態で知識コードを管理しておき、この設計手順書と知識コードとの変換を行えばよい、と考える。具体的には次の3つのタスクを実現することを考える。

1. 設計手順書の日本語テキストから知識コードへの変換。
2. 異なるCAD間での知識コードの相互変換。
3. 日本語テキストの形態で知識コードを管理できるよう、CAD知識コードを設計手順書へ変換。

本研究で開発した知識コードコンバータC³は、この3つのタスクを実現することにより知識ベースCAD導入によって生じる問題点を解決し、その活用を促進するものである。

3. 知識コードコンバータの構成

3.1 中間表現コード

前節で述べた3つのタスクを実現するために、本システムではまず中間表現コード(IKC; Intermediate Knowledge Code)の概念を導入する。中間表現コードは各知識ベースCAD毎に異なるコード仕様に依存せずに知識を表現することを可能とする中間コードである。中間表現コードを導入するためには、設計知識を表現するための一般的な概念と個別のCADに固有の概念を明確に区別して整理することが必要である。一方で中間表現コードを用いて一般的な概念とCAD固有の概念とのマッピングをとることにより、比較的簡単にC³に新たな知識ベースCADを加えることが可能となる。

中間表現コードと類似した試みは既存の研究に見ることができる。例えば幾何形状表現や製品情報に関しては、個別のCADに依存しない一般的な表現方法としてSTEP⁷⁾が提案されている。また知識共有に関する研究分野においては、個々のアプリケーションに依存しない知識記述フォーマットとしてKIF(Knowledge Interchange Format)⁸⁾が提案され、一定の成功を収めている。

これらに対して本研究で定義する中間表現コードは、幾何形状への操作の表現を基本に、操作手順や設計ルール、制約条件を含む設計知識を記述するための一般的な枠組みを構築するものであり、この点で新しい概念である。本研究では、複数の異なるCADシステムの操作に習熟している設計者との議論を通じ、「2点を通る直線を作成」、「平面をオフセット(平行移動)」などといった一般的なCAD操作を51種挙げ、中間表現コードとして定義した。表2に本研究で定義した中間表現コードの例を示す。中間表現コードは、操作名、および

Table 2 Examples of IKC

Type of IKC	Body
Conditional	(#conditional ((#condition <Operation>) (#result <Operation>)))
Line by 2 points	(#create_line ((#point1 <Point>) (#point2 <Point>)))
Offset plane	(#create_offsetplane ((#object <Plane>) (#direction <Direction>) (#distance <Value>)))

Table 3 Correspondence of processes and tasks of C^3

Task	Function			
	I	II	III	IV
1. Generation of CAD knowledge code	X	X		
2. Mutual exchange of CAD knowledge code		X	X	
3. Generation of Design Procedure Document			X	X

Fig. 3 The architecture of C^3

複数の操作引数名 (表中, #を付された語)・引数変数 (<> で記された部分) により構成される。引数変数は幾何情報を参照するための変数である。引数変数の値として全角文字を取ることが可能であるが, 操作名および操作引数名には計算機の内部処理上の利便性を考慮して半角英数記号を用いる。なおここで定義した操作の数は最終的なものではなく, 今後システムの検証を進めていく過程で更新される可能性がある。

3.2 C^3 の機能

中間表現コードの概念を導入することにより, 前節で挙げた C^3 の 3 つのタスクは次の 4 つの機能で実現される。

- I. 設計手順書 (DPD; Design Procedure Document) に記述された日本語テキストを中間表現コードに変換する。
- II. 中間表現コードを CAD に固有の知識コード (CAD 知識コード, CKC; CAD Knowledge Code) に変換する。
- III. CAD 知識コードを中間表現コードに変換する。
- IV. 中間表現コードを設計手順書の日本語テキストに変換する。

4 つの機能と 3 つのタスクとの対応関係を表 3 に示す。表 3 中の "X" は, 機能とタスクの対応関係を示している。

3.3 C^3 のコンポーネント

4 つの機能の各々を実現するために, C^3 のコンポーネントとして, DPD-IKC 変換器, IKC-CKC 変換器, CKC-IKC 変換器, IKC-DPD 変換器の 4 つを考える。これらのコンポーネントで構成される C^3 のアーキテクチャを図 3 に示す。図 3 中, ローマ数字付き矢印が機能, 灰色の長方形ノードがコンポーネントをそれぞれ示す。

以下, 各々のコンポーネントについて説明する。

3.3.1 DPD-IKC 変換器

DPD-IKC 変換器は, 日本語で記述された設計手順書の内容を中間表現コードに変換するコンポーネントである。ただ

Table 4 Design Procedure Document

No.	Ref. Name	Body
B.	箱内面・分割位置の設計手順書	
B1	箱下面	<メディア下面> と同位置に平面を作成する。
B2	箱上面	<メディア上面> から 5mm 外側にオフセット平面を作成する。
B3	箱左面	<メディア左面> から 3mm 外側にオフセット平面を作成する。
B4	箱右面	<メディア右面> から 3mm 外側にオフセット平面を作成する。
B5	箱前面	<メディア前面> から 5mm 外側にオフセット平面を作成する。
B6	箱後面	<箱前面> から <箱後面の長さ> 外側にオフセット平面を作成する。

Table 5 Example of DPD to IKC conversion rule

Condition :
<Plane> から <Value>mm<Direction> にオフセット平面を作成する。
Generation :
(#create_offsetplane ((#object <Plane>) (#direction <Direction>) (#distance <Value>)))

し, 本変換器で処理する設計手順書は, 表 4 に示す形式で記述されている必要がある。この形式における設計手順書の 1 行が 1 つの記述に対応しており, 各行は以下の 3 つのスロットを持つ。

行番号 設計手順書の 1 行ごとに付けられる通し番号である。

参照名 記述本体に記述された手順を実行した結果, 生成される幾何情報に対する参照名である。設計手順書内において他の幾何情報を参照する際にはこの参照名を用いることができる。

記述本体 記述の内容を示す日本語のテキスト情報である。設計手順, 設計ルール, 設計制約を記述することができる。

設計手順書の日本語記述と中間表現コードの変換を行うに当たり, その変換規則を記述した DPD-IKC 変換ルールを定義する。このルールは, 表 5 に示すように条件部に設計手順記述の構文情報, 生成部に中間表現コードの構文情報を持つ。本変換器は, 設計手順書の各行の構文と条件部の構文とがマッチする DPD-IKC 変換ルールを探索し, 中間表現コードを生成する。例えば「箱上面から 5mm 上方にオフセット平面を作成する。」という記述は表 5 に示したルールにマッチし, "(#create_offsetplane ((#object 箱上面) (#direction 上方) (#distance 5)))" という中間表現コードが生成される。

構文にマッチする DPD-IKC 変換ルールが見つからなかった場合, DPD-IKC 変換器は設計手順書の記述に誤りがあると判断し, ユーザに記述の修正を要請する。

上記の DPD-IKC 変換器における処理のアルゴリズムを図 4 に示す。

なお本研究では構文解析時において形態素解析ツール「茶筌 version2.1」⁹⁾ を使用している。

3.3.2 IKC-CKC 変換器

IKC-CKC 変換器は, 中間表現コードを各 CAD 知識コードに変換するコンポーネントである。本論文では知識ベース CAD として CATIA V5 を利用して議論を進めるが, ここで提案する枠組みは他の知識ベース CAD に対しても同様に適用可能なものである。

CKC への変換にあたり, 表 6 に示すように条件部に中間表現コードの構文情報, 結論部に CAD 知識コードの構文情報を持つ IKC-CKC 変換ルールを用いる。このルールは知識ベース CAD に対して個別に用意される。本生成器は, 入力された中間表現コードを 1 行ずつ取り出してその構文を調べ, 変

Fig. 4 Algorithm of IKC generation

Table 6 Example of IKC to CKC conversion rule

Condition :
(#create.offsetplane ((#object <Plane>) (#direction <Direction>) (#distance <Value>)))
Generation :
Dim ?parameter As Parameters Set ?parameter = &part.Item("<Plane>") Dim ?offsetplane As HybridShapePlaneOffset Set ?offsetplane = &factory.AddNewPlaneOffset(?parameter, <Value>,<Direction>)

換ルールの条件部にマッチするか否かをチェックする。このときに本変換器は、幾何情報に関して中間表現コードの中に残されている日本語記述と各知識ベース CAD の内部表現との対応関係を管理しておき、適切な表現に変換して CAD 知識コードを出力する。

例えば表 6 には、"(#create.offsetplane ((#object 箱上面) (#direction 上方) (#distance 5)))" という中間表現コードがマッチし、最終的に表 7 に示す CAD 知識コードが生成される。このコード中の「箱上面」や「上方」の記述が、それぞれ「A1」(「箱上面」オブジェクトを指す変数)、「False」(「箱上面」オブジェクトの「上方」を示す内部表現)という表現に変換されている。

3.3.3 CKC-IKC 変換器

CKC-IKC 変換器は、CAD 知識コードを中間表現コードに変換するコンポーネントである。変換にあたっては、条件部に CAD 知識コードの構文情報、生成部に中間表現コードの構文情報を持つ CKC-IKC 変換ルール (表 8 参照) を利用する。CKC-IKC 変換ルールは個別の知識ベース CAD に対してそれぞれ用意する必要があるが、本変換器のアルゴリズムそのものはどのような知識ベース CAD に対しても利用可能である。

本変換器と前述した IKC-CKC 変換器により C^3 は、複数の知識ベース CAD の間における CAD 知識コードの相互変換を実現することができる。

3.3.4 IKC-DPD 変換器

IKC-DPD 変換器は、中間表現コードを設計手順書の日本語記述に変換するコンポーネントである。変換にあたっては、条件部に中間表現コードの文法情報、生成部に設計手順書の文法情報を持つ IKC-DPD 変換ルール (表 9 参照) を利用す

Table 7 Generated CKC

Dim hybridShapeSurfaceExplicit1 As Parameters Set hybridShapeSurfaceExplicit1 = part1.Item("A1") Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset Set hybridShapePlaneOffset1 = factory1.AddNewPlaneOffset(hybridShapeSurfaceExplicit1, 5, False)

Table 8 Example of CKC to IKC conversion rule

Condition:
Parallel.Name = <name>, Mode = <mode>, Type = <type>, Curve = <curve>, Support = <support>, Offset.Mode = <offsetmode>, Length = <length>, Bothside = <bothside>, Direction = <direction>
Generation:
((create.trim_line (((name)(<name>)) ((object1)(<curve>)) ((object2)(<support>)) ((direction)(<direction>)) ((distance)(<length>))))))

Table 9 Example of IKC to DPD conversion rule

Condition:
((create.trim_line (((name)(<name>)) ((object1)(<curve>)) ((object2)(<support>)) ((direction)(<direction>)) ((distance)(<length>))))))
Generation:
Ref.Name:<name> Body: <curve> から <length>mm <direction> に <support> の支持のもとで オフセット曲線を作成する。

る。本変換器は前述した CKC-IKC 変換器と共に利用することで、設計手順書の生成を実現する。

4. Maxis C^3 の開発

4.1 実装

以上の議論に基づき、Maxis C^3 の開発を行った。開発言語は C++ を利用し、プラットフォームは Windows2000 である。現行の開発バージョンが対応する知識ベース CAD は CATIA V5 である。

4.2 実行例

現行開発バージョンの Maxis C^3 は、設計手順書の日本語記述を CATIA V5 用の知識コードに変換すると共に、CATIA V5 の知識コードを設計手順書の日本語記述に変換することが可能である。本節では Maxis C^3 の機能を示すために、図 5 に示すメディアボックスの設計を CATIA V5 上で行った事例を用いて説明する。

CATIA V5 上で設計を行うにあたり、設計者はメディアボックスの設計手順書を作成した。表 4 に示した設計手順書はその一部である。この設計手順書を Maxis C^3 に入力すると、Maxis C^3 は図 6 に示すように DPD-IKC 変換器を用いて設計手順書の日本語記述 (図 6 上段) を中間表現コード (図 6 下段) に変換する。

中間表現コードが生成されると、Maxis C^3 は図 7 に示すように CATIA V5 用の IKC-CKC 変換器を用いて CATIA V5 の CAD 知識コード (図 7 下段) を生成する。生成された CAD 知識コードは CATIA V5 へ入力され、図 5 に示す CAD モデルを自動的に生成する。

一方、CAD モデルを CATIA V5 で作成した後に Maxis C^3 を用いると、CATIA V5 が記録した設計手順のコードをもとに、IKC-IKC 変換器および IKC-DPD 変換器により日本語で記述された設計手順書を生成する (図 8 参照)。

以上に示したように、Maxis C^3 は日本語で記述された設計手順書を知識ベース CAD の知識コードに変換するとともに、知識ベース CAD 内部に記録された知識コードから日本語の設計手順書を生成することが可能である。すなわち Maxis C^3 により知識ベース CAD の知識コードを日本語の設計手順書の形態で作成し、管理することが可能であり、Maxis C^3 のこの機能は、知識ベース CAD を今後一層の活用するために有用であると考えられる。

4.3 検証

C^3 の効果を検証するために、2. 章で述べた自動車部品設計

Fig. 5 Media box

Fig. 6 Convert DPD to IKC

Fig. 7 Convert IKC to CKC

会社において、知識ベース CAD と C^3 を同時に導入した際の作業量の変化を調査した。ただし次年度以降のデータについては、初年度に現れた導入効果および表 1 に示した調査結果を元に推測した。表 10 にその調査結果を示す。

知識ベース CAD と C^3 の導入初年度には、CAD モデリング作業が大幅に削減されるものの、知識コード化に費やされる作業量が大きいため (25)、導入前と比較すると作業量合計は増加する (32)。しかし知識ベース CAD を単独で導入した場合と比較すると、その増加幅は小さくなっている (表 1 参照)。これは C^3 によって知識コードの作成・保守を日本語ベースで行うことができるため、作業の負担が軽減されるからである。知識ベース CAD と C^3 の導入次年度以降では、表 1 に示した調査から、知識コードの作成・保守に費やされる作業量を大幅に削減できると予想される。その結果、作業量の合

Fig. 8 Generation of design procedure document

Table 10 Variation of loads by the introduction of C^3

	Load of designer (ManDay)*			Sum.
	(a)	(b)	(c)	
Before Knowledge-based CAD	24	-	-	24
1st year after introducing Knowledge-based CAD + C^3	2	25	5	32
Afterward	2	5	3	11

* (a): CAD modeling, (b): Knowledge encoding, (c): Knowledge maintenance

計は知識ベース CAD 導入以前の半分以下となり、費用対効果に見合う結果となる。

以上の考察により、 C^3 は知識ベース CAD 活用のために大きな役割を果たし、知識ベース CAD の導入効果を高めることが分かった。

5. 結 論

本論文では、設計知識管理の実現に向けて知識ベース CAD を効果的に活用するために開発した Maxis C^3 の枠組みとその実装状況について紹介し、その有効性を示した。なお本システムの基本的なアイデアは、日本および米国において特許出願中 (特願 2002 - 338832, 特願 2003-305667, Serial No.: 10/716,557) である。

参 考 文 献

- 1) Dieng, R.: Knowledge Management and the Internet, IEEE Intelligent Systems, **15**, 3 (2000) 14.
- 2) Mekhilef, M. and Deshayes, P.: "Knowledge Management: A Concept Review," Proceedings of the ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, (2003), DAC-48745 (in CD-ROM).
- 3) Yoshioka, M. and Shamoto, Y.: "Knowledge Management System for Problem Solving - Integration of Document Information and Formalized Knowledge -," Proceedings of the ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, (2003), CIE-48217 (in CD-ROM).
- 4) 設計知識を 3 次元 CAD に組み込める知識ベースモデリング機能, 日経 CG, **162**, (2000), 106.
- 5) 園川隆夫, 安達俊行: "製品開発論", 日科技連出版社, (1997).
- 6) Roth, F. H., et al.: "Building Expert Systems," Addison-Wesley Publishing Company, Inc., (1985).
- 7) 木村文彦 ほか, "特集プロダクトモデルと CAD データ交換標準 STEP", 精密工学会誌, **59**, 12, (1993), 1917.
- 8) Genesereth, M. R.: "Knowledge Interchange Format", Proceedings of the Conference of the Principles of Knowledge Representation and Reasoning, Morgan Kaufmann Publishers, (1992), 599.
- 9) 松本裕治 ほか: 日本語形態素解析システム『茶釜』使用説明書, 松本研究室, 奈良先端科学技術大学院大学, (1999).