

TR-455

設計問題向けツール・アーキテクチャ

永井 保夫(東京)、滝 寛和、
寺崎 智、横山 孝典、井上 克巳

April, 1989

©1989, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

設計問題向けツール・アーキテクチャ

A Tool Architecture for Design Expert Systems

永井 保夫* 滝 寛和 寺 崎 智
Yasuo NAGAI Hirokazu TAKI Satoshi TERASAKI
横山 孝典 井上 克巳
Takanori YOKOYAMA Katsumi INOUE

(財) 新世代コンピュータ技術開発機構
Institute for New Generation Computer Technology

Keywords: design expert system, tool architecture, constraint compilation

1. はじめに

設計向けの知識ベースシステムを開発していくためには設計対象に依存した知識が大量に必要とされる。従って、設計問題固有の知識表現と、その知識を用いた効率の良い問題解決の方法が設計システム向けのツール研究におけるキーポイントとなっている。本研究では、このような設計問題を対象にした知識ベースシステムのアーキテクチャを明確にして、構築ツールを実現することを目的としている。

ところで、設計における標準的な部品・手法や設計に関するノウハウなどは、対象に深く依存しているため個別性が非常に高い。また設計システムを実現するために必要な設計作業の手順化も一部の定型化されている部分を除いて十分には行われていない。多くの設計システムはこのような理由によって、特定の用途に応じた専用システムとして構築・利用されている。このアプローチにより構築されたシステムは個性が強いため、その保守および拡張が困難であり、システムの汎用的な利用という点からも非効率である。従って、システム構築者（設計者を含む）が設計システムを構築・利用できる汎用的なツールの実現が要求されている。つまり、従来のように設計知識を手続き的プログラムとして表現したり、特定の問題解決機構に依存した形式での記述を強いることは避けなければならない。これは、設計知識がシステム内部に埋め込まれることにより設計者から見えなくなることを避け、設計知識の再利用や設計者間での継承を容易にするという点からも重要なことである⁽⁸⁾。

そこで我々のアプローチは、Fig. 1 に示すように設計知識と問題解決機構を独立させ各々宣言的に記述し、そのうえで両者を結合してシステムが構築できる環境をツールとして提供するとともに、構築されたシステムの問

題解決機構の効率化をはかるものである。

設計知識は対象間の関係を数式などの記号によって表現している場合が多い。設計知識を手続き的表現を用いて記述した場合と宣言的な表現を用いて記述した場合を比較すると、後者の方が記述能力が高くシステム構築者にとってはより自然であると考えられる。つまり、このような関係を表現する場合、手続き的表現では関係を満足させる方法を陽に与える必要があるのに対して、宣言的表現では、その必要がなくなるという利点がある。

ここでは、設計知識の宣言的表現を容易にするために対象間の関係を制約として表現する。これにより、従来の手続き的な知識表現と比較してより高度な知識表現が実現でき、設計向け知識ベースシステムの構築に有効であると考えられる。

一方、制約をうまく利用することにより、探索すべき解空間を制限できるため、余分な探索を減らし問題解決の効率を上げることも期待できる。

我々は、このような制約表現の利点に注目し、設計問題向けの制約問題解決の実現を検討しており、その最重要技術課題として知識コンパイラを考えている。知識コンパイラは、一般的には物理法則や構造などの原理的な知識（深い知識）からより実行効率の良いタスクに直結した知識（浅い知識）を生成する知識変換技術をいう⁽¹³⁾

これに関して、特に我々は制約に着目した知識コンパイラを実現し、これを制約指向知識コンパイラと呼んでいる。制約指向知識コンパイラを用いることにより、宣言的に表現された設計知識から効率的な設計手順の自動生成が可能となる。

本稿ではまず、2 章において知識表現と問題解決機構を検討課題として取り上げる。次いで、設計問題向けツ

* 現在、(株) 東芝 情報通信システム技術研究所に所属。

ールを検討していくために考慮すべき項目として、3 章では対象モデル、4 章では設計過程モデルおよび制約問題解決機構について述べる。最後にツール全体のアーキテクチャについて考察し、その構成要素である制約指向知識コンパイラ及び対象モデラを紹介する。

2. 設計問題向けツールの検討課題

2. 1 知識表現

設計知識は、設計対象そのものに関する知識と、設計対象を解析・詳細化し設計解を求めるための知識に大別される。前者は設計対象の構造、形状、属性などを表現したもので、従来から対象モデルと呼ばれている⁽⁷⁾⁽¹⁰⁾。後者の対象モデルの解析方法や設計手順に関する知識をここでは設計手法知識と呼ぶことにする。すなわち、設計手法知識は対象モデルの利用法や設計パラメータの値の求め方、設計手順、解候補が複数あった場合の探索手順、設計要求を満たさない時の解の補正操作、および解の評価に関する知識等からなる。これらの知識には、関数や不等式で表されたもの、カタログや図表の検索に関するものなどがあり、さまざまな種類や表現形式を持つ。さらに、設計公式やカタログなどの汎用的な知識と、ある特定目的に対してのみ有効なヒューリスティクスが混在している。

構築ツールの知識表現を考えるにあたっては、これらの役割や性質に応じた整理が必要である。特に、対象モデルと設計手法の知識の明確な分離が重要である。従来のシステム⁽²⁾⁽³⁾では、これらの違いが明確に意識されておらず、対象モデルとして表現すべき知識が設計手続き中に埋め込まれている場合が多い。しかしこれでは対象に関する知識と設計手法に関する知識が未分化になり、システムの柔軟性を損ない、設計システムの高度化や汎用性のあるシステムの構築を阻害してしまう。知識表現を分離することで、4 章で述べるように設計知識を設計過程モデルとして自然な形で表現でき、設計のコンテキストを明確にした知識の再利用が容易となる。

また、多くの設計知識は特定の対象に依存したヒューリスティクスから成り立っているが、できるだけ共通化して利用する必要がある。そのため、設計知識を取り扱う際には、標準部品や公式・実験式などの一般的な知識を予め用意しておくことにより知識の記述量を減らすことが望まれる。5 章のツール・アーキテクチャでは、このような点を考慮した対象モデラおよび知識ベース管理機構について示す。

2. 2 問題解決機構

一般に設計問題では、不等式のように解の生成範囲を規定するだけで解法が陽に与えられない場合が多い。

従って、設計過程を制約充足過程と捉え、制約問題解決の枠組みを適用することが有効であると考えられる。

さらに、対象モデルは設計対象の構造の制約を表すのみならず、また要求仕様及び関数、(不)等式あるいはカタログや図表の検索に関する知識(設計手法に関する知識)も対象間の関係を表現した制約とみなすと、設計過程全体を制約問題解決の観点から統一的に捉えることが可能となる。ただし、現実の設計で扱われる制約条件は、性能と納期・コストといった条件間のトレードオフや、設計者の意図・好みにより優先度が付けられたり動的に変更されるなど、単純に扱うことができない場合が多い。

また、設計問題を対象とした場合、従来のエキスパートシステム構築支援ツールでは制約を宣言的に表現できるものは少なく、システム構築者自身がツールの開発言語を駆使して特定の問題に適した制約問題解決機構を実現していることが多い。

従って、設計問題向けツールを考えるにあたっては、制約を明示して宣言的に表現し、それに適した問題解決機構を実現する必要がある。そのためには、まず設計問題における制約の概念を整理することが重要である。我々は、制約の性質を以下のように捉えている⁽⁵⁾⁽⁸⁾

a) 静的な制約と動的な制約

多くの制約解決機構では、制約を一定不変(静的)なものとして扱っている。ところが、現実の設計問題では必要となる全ての制約条件が予め明らかになっているとは限らず、設計が進むにつれて追加・削除され、動的に変化することが多い。

b) 義務的な制約と示唆的な制約

設計問題ではすべての制約を対等に扱うわけではない。つまり、制約には優先度があり、その重要性は設計要求や設計者の意図に依存する。義務的な制約は常にすべてが満足されねばならず、一般的には陽に与えられる。一方、示唆的な制約は、可能な選択枝から最良の枝を選択するためのガイドとして使用されるのが一般的で、義務的な制約に比べて優先度は低い。従って、義務的な制約を満足させるために、示唆的な制約を変更する場合もありうる。

c) 局所的な制約と大局的な制約

設計では問題をいくつかの部分問題に分割して解くことが多い。この場合、適用範囲が問題全体に及ぶ大局的な制約と、部分問題の中だけに適用される局所的な制約を区別する必要がある。さらに、局所的な制約充足だけで大局的な制約充足が実現できるかについても考慮する

必要がある。

d) 有限領域と無限領域

制約伝播は、制約を満足させるように制約中の変数（制約変数）がとりうる値を決定する手法である。この場合に変数がとりうる値の領域には、有限領域と無限領域がある。これらを扱う方法としては値の伝播と値が取り得る区間の伝播を考慮する必要がある。

以上、設計問題における制約の分類を行ったが、実際にはひとつの制約が上記の複数の性質を有し、これらを可能な限り統一的に扱う枠組みが望まれる。4章ではそのなかのいくつかの性質を考慮した制約問題解決機構について検討している。

3. 対象モデル

設計対象に関する知識を表現する場合、対象モデルの概念が重要になる。モデルとは現実の対象を問題解決機構やユーザが取り扱いやすい形式で表したものである。設計とは、このモデルを詳細化していく過程であるとみなされる。

このような対象モデルの表現については、オブジェクト指向による表現が主流で数多くの適用例がある⁽¹¹⁾。

しかし、従来のオブジェクト指向言語は制約を表現する機能が乏しく、付加手続きやメソッドにより手続き的に記述する必要がある。そのままでは対象モデル記述用の知識表現として優れているとは言い難い。そこで、オブジェクト指向に制約を導入する方式が有力と考えられる。

4. 設計過程モデルと制約問題解決機構

設計過程を制約充足過程と捉えて、設計問題向け問題解決機構を実現するためには、設計過程モデルを設定することが必要である。

そこで我々は、まず設計過程をFig. 2のように、設計プランに基づいて問題分割と詳細化を繰り返す問題解決的な処理としてモデル化した。さらに、このモデルに基づいた制約問題解決機構について検討を進めている。

ここでは、各部分問題における設計過程を値を生成するジェネレータ、制約伝播を行うプロパゲータ、テスト、および失敗回復処理などの問題解決プリミティブから構成されたモデルとみなし、制約問題解決を行う。

Fig. 3のモデルは、生成検査法を拡張させたもので、ジェネレータによりパラメータに値が割り付けられると、その値を入力とみなして、プロパゲータが制約を能動的に解釈し、局所的な処理や大局的な処理を行なう。テストは制約内のすべての変数に対して、値が割り付けられると、制約を受動的に解釈し、その評価を行なう。なお、

ここで能動的な解釈とは、特別な解法を用いて、積極的に制約を解くことであり、受動的な解釈とはジェネレータやプロパゲータにより得られた値が制約を満足しているかを検査するものである。失敗回復処理は、プロパゲータによる能動的な値の割り付けやテストによる制約評価が失敗した場合に再実行を行う。

この失敗回復処理で単純なバックトラックだけを用いたのでは全体の処理効率が落ちるので、手続き的な経験的知識を使って、生成検査法を効率的に実行したり、制約の緩和を行なう。

実際には、ジェネレータの実行順序、つまり、変数に対して値を割り付ける順序によって、制約伝播の実行順序が変わるため、問題解決の処理手順も変化する。そこで、コントローラは、問題解決の効率を向上させるために、各問題解決プリミティブの制御を行なう。

5. 設計問題向けツール・アーキテクチャ

我々はこれまでに設計知識を対象モデルと設計手法知識に分離して取り扱うことで、知識の表現力を高め、その結果システム全体の処理能力が改善し得ることを示した。

またこれらの知識や与えられる要求仕様を制約と捉えることにより、設計問題を制約問題解決という統一的な視点に基づいて解く問題とみなすことができる。

このように分離された設計知識を統一的に取り扱い、4章で述べた制約問題解決機構のプリミティブに基づいて制約（設計知識）を設計手順へ写像し、問題解決の効率化をはかるためには、知識変換技術が重要である。

現在、この知識変換技術を用いた知識の有効な活用法として、次の二つの方法を検討している。一つは1章に述べた知識コンパイルによる方法で、対象モデルに記述された知識と設計手法に関する知識から設計手順を自動生成し、設計システムを作成する方式である。これは特に対象の構造がほぼ固定されているパラメトリック設計において有効で、効率よい問題解決を実現できる⁽⁴⁾⁽⁵⁾。

もう一つの方法は、問題解決時に対象モデルに記述された知識と設計手法知識を中間表現に変換しておき、その中間表現を順次解釈しながら設計を支援するものである。これはインタプリタを用いて動的に設計手順を生成しながら、解釈することに相当する。さらに、このような変換技術によって過去に得られた知識（設計手順やその中間表現）を再利用することによって、問題解決時に生成される設計手順を効率的に解釈することが可能となる。

我々は第一段階として、あらかじめ設計対象の構造や設計手法知識が明確になっている設計問題を取り扱った。従って、知識変換技術としては、知識コンパイル方式を採用した。

以下では、まず始めにツールのアーキテクチャの基本構成要素を述べる。つぎに、そのなかで設計知識を統一

的に取り扱い、問題解決の効率化を図ることを目標とした制約指向知識コンパイラについて紹介する。さらに、対象モデルを容易に記述できることを目指した対象モデル（設計対象表現システム）についても紹介する。

5. 1 ツール・アーキテクチャ

我々の考えているツール・アーキテクチャは Fig. 4 に示され、その基本構成要素は、ユーザ・インターフェイス、対象モデル、制約指向知識コンパイラ、制約問題解決機構、知識ベース管理機構からなる。

ツールへの入力としては、ユーザ・インターフェイスを介して、対象モデルと対象モデルの属性を詳細化するための設計手法を与える。

対象モデルは、基本的な部品や機能ブロックを知識ベース管理機構側でライブラリとして用意しておき、これらを参照・変更することにより設計対象を表現する。

設計手法知識も対象モデルと同様に、一般に良く知られた公式や実験式、及びカタログや図表の検索方法に関する知識をライブラリ化し利用する。過去の設計事例の参照や設計解の探索に関するヒューリスティクスなども設計手法として表現する。

制約指向知識コンパイラは、設計知識を有効に生かして、効率的な問題解決を行うシステムを構築するために用いられる。

対象モデルは、問題解決器やユーザ自身による対象モデルの修正や詳細化などの処理を可能にするための表現機構である。

以下では、我々がアーキテクチャの主要構成要素のなかで、特に重要であると考えている制約指向知識コンパイラおよび対象モデルについて述べる。

5. 2 制約指向知識コンパイラ

制約指向知識コンパイラは設計手順を生成するという意味では知識コンパイラ⁽¹⁾と同様であるが、以下の特徴を有している。

第1に、制約指向知識コンパイラは設計知識をすべて制約とみなすことにより、さまざまな形式で表現された知識を統一的に扱うことができる。言い替えれば、ノウハウやヒューリスティクスといった領域に依存した知識も制約と見なし、これらを汎用的に用いられる知識と組合わせて、ある特定目的のための特化した知識に変換するものといっても良い。第2に、宣言的に表現された設計知識（制約）の使い方に関する知識を与え、データフロー情報を用いて制約間の依存関係を解析し、手順を生成することができる。第3に、制約間の依存関係に基づいて、あらかじめ設定された問題解決機構に適した設計手順を生成するので、効率的に実行できる。これにより、設計知識と問題解決機構との結合機能を提供できるので、設計知識を問題解決機構とは独立に表現できる。

第4に、与えられた設計知識の間の依存関係を設計者に示すことにより、設計知識の修正を支援できる。

作成された設計システムは生成された設計手順に従って、4章で示された設計過程モデルに基づき制約問題解決を行い、要求仕様を満足する設計解を求める。

我々は、このような制約指向知識コンパイラを適用した設計支援システム構築ツール MECHANICOT を開発した⁽¹²⁾。MECHANICOT は機械のパラメトリック設計⁽⁹⁾を対象に、設計対象の構造やパラメータ間の依存関係に着目して設計手順を生成し、専用の設計システムを出力するツールであり、そのアーキテクチャは Fig. 4 とほぼ同様な構成である。

5. 3 対象モデル（設計対象表現システム）

MECHANICOT では、構造があらかじめ固定された設計対象だけを取り扱っているが、実際の設計では構造が固定されておらず、動的に修正される場合が多い。このように設計支援システムをより実際的なシステムとして実現するためには、設計対象の構造を動的に修正可能とする必要がある。たとえば、知識コンパイラが難しく試行錯誤的に問題を解く必要がある場合や、ユーザと対話的に処理を進める場合に、有効な知識の利用法としては、インタプリタ方式が有力である。

また MECHANICOT では、対象モデルを単なる対象モデルの入れ物として実現している。ところが、上記のような実際的なシステムに対処するためにはより高機能なモデルが必要とされるため、問題解決機構を有したモデルの検討も行っている。具体的には、対象モデルに記述された制約や動的に付加された制約を解釈し、設計過程で生成される解を常に制約充足状態に保つ機能を提供し、効率のよい設計支援を行うことを検討し、そのための設計対象表現システム FREEDOM を開発中である⁽¹⁴⁾。

6. おわりに

以上、設計問題向け知識ベースシステム並びにツールの検討課題及びそのアーキテクチャについて、知識表現と問題解決という二つの側面から論じた。次に、対象モデル、設計過程モデル及び問題解決機構を制約という観点から検討し、その実現に要求される機能について述べた。

そして、知識の表現力の向上を目的として設計知識が統一的に取り扱え、かつ問題解決の効率化が可能なシステムを構築するためのツール・アーキテクチャについて論じた。特に、そのなかで重要となる制約指向知識コンパイラ並びに対象モデルについて紹介した。

今後は、設計問題を題材としたより実際的なシステムを実現するために、制約指向推論、仮説推論、及び知識コンパイルなどの知識変換技術との関連をより明確にし、

個々の技術を統合化することにより、全体システムの処理の高度化を図る計画である。さらに、処理の高速化のための並列化についても検討していくことを考えている。

謝辞

本研究を行うにあたり、ICOTの昭和62年度設計型エキスパートシステムサブワーキンググループ（長澤主査）での議論が大いに参考になった。主査並びに委員の方々に深く感謝致します。また、設計問題の具体例の提供、問題の解析にあたり多大な御援助を頂いた工業技術院機械技術研究所の小島氏、今村氏に厚く御礼申し上げます。また、本研究の機会を与えて下さり常に御指導頂いたICOTの淵所長、第5研究室の藤井室長に深く感謝致します。

<参考文献>

- (1) Araya, A. A. and Mittal, S., Compiling Design Plans from Descriptions of Artifacts and Problem Solving Heuristics, Proc. of IJCAI87, (1987)
- (2) Brown, D. C. and Chandrasekaran, B., Knowledge and Control for a Mechanical Design Expert System, IEEE Computer pp.92-100, July (1986).
- (3) Mittal, S. and Dym, C. L. and Morjaria, M., PRIDE - An Expert System for the Design of Paper Handling System, IEEE Computer, July pp.102-114, (1986).
- (4) Nagai, Y., Towards Design Plan Generation for Routine Design Using Knowledge Compilation - Focusing on Constraint Representation and Its Application Mechanism for Mechanical Design -, ICOT Technical Memorandum, TM-504, (1988).
- (5) Nagai, Y., Towards an Expert System Architecture for Routine Design - Focusing on Constraint Representation and an Application Mechanism for Mechanical Design -, ICOT Technical Memorandum, TM-456, Proc. of 3rd Int'l Conf. on CARS & POF88, (1988).
- (6) Nagai, Y., Terasaki, S., Yokoyama, T., and Taki, H., Expert System Architecture For Design Tasks, Proc. of Int'l Conf. on FGCS88, ICOT, (1988)
- (7) Ohsuga, S., Conceptual Design of CAD Systems Involving Knowledge Bases, Knowledge Engineering in Computer-Aided Design, (Gero, J. S. (ed)), North-Holland, pp.29-50, (1985).
- (8) 昭和62年度 XSS-WG, DBS-SWG 報告書, 設計型エキスパートシステム研究に対するイメージ及び技術課題について, 永井 保夫(編集), ICOT Technical Memorandum, TM-681, (1989)
- (9) 井上 克巳, 永井 保夫, 藤井 裕一, 今村 聡, 小島 俊雄, 工作機械の設計手法の解析-旋盤の回転機能部品の設計, ICOT Technical Memorandum, TM-494, ICOT, (1988)

- (10) 上野 晴樹, 知識工学入門, オーム社, (1985)
- (11) 木村 文彦, オブジェクト指向による CAD/CAM のためのモデリングとデータベース, 情報処理, Vol. 29, No. 7, (1988) pp.368-373.
- (12) 寺崎 智, 永井 保夫, 横山 孝典, 井上 克巳, 堀内 英一, 滝 寛和: 機械設計支援システム構築ツール - MECHANICOT -, 人工知能学会, 知識ベースシステム研究会資料, SIG-KBS-8803, (1988).
- (13) 山口 高平, 溝口 理一郎, 田岡 直樹, 小高 浩, 野村 康雄, 角所 収, 深い知識に基づく知識コンパイルの基本設計, 人工知能学会誌, Vol. 2, No. 3, pp.77-84, (1987)
- (14) 横山 孝典, 設計対象記述のための知識表現システム FREEDOM の提案, 情報処理学会, 知識工学と人工知能研究会資料, 88-60, (1988).

Expert systems for design problems

Expert system building tool for design problems

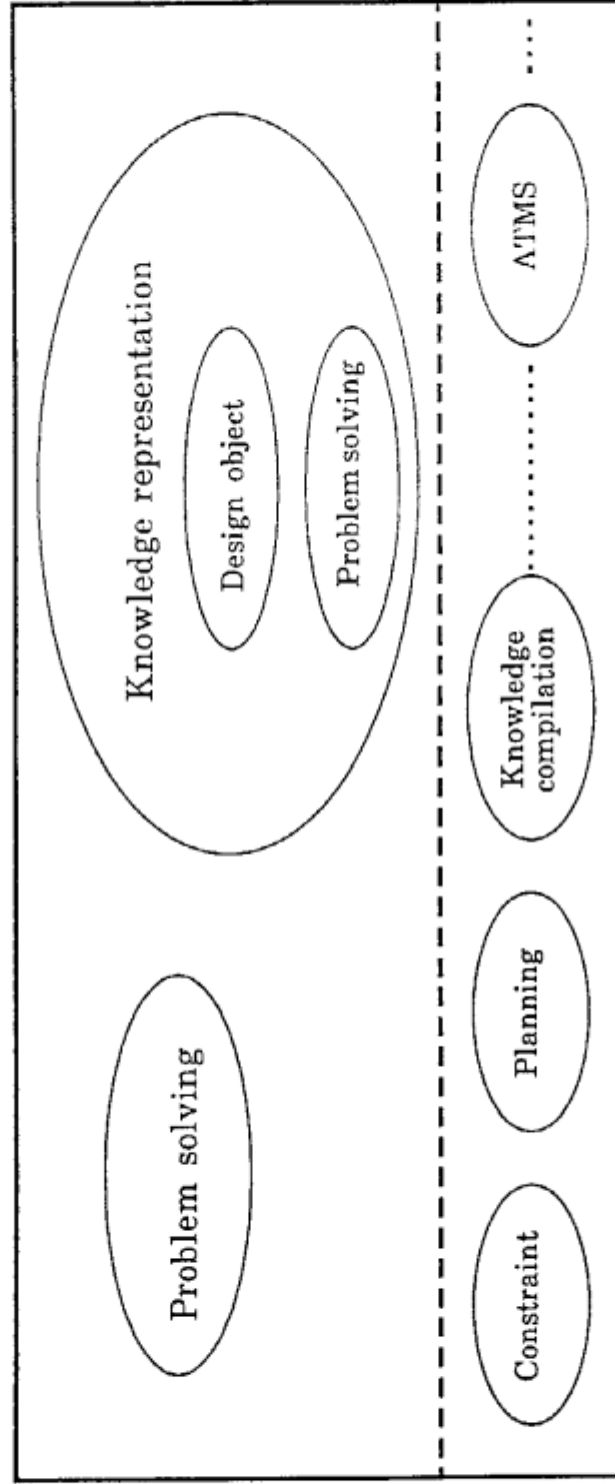


Fig. 1 Research Approach

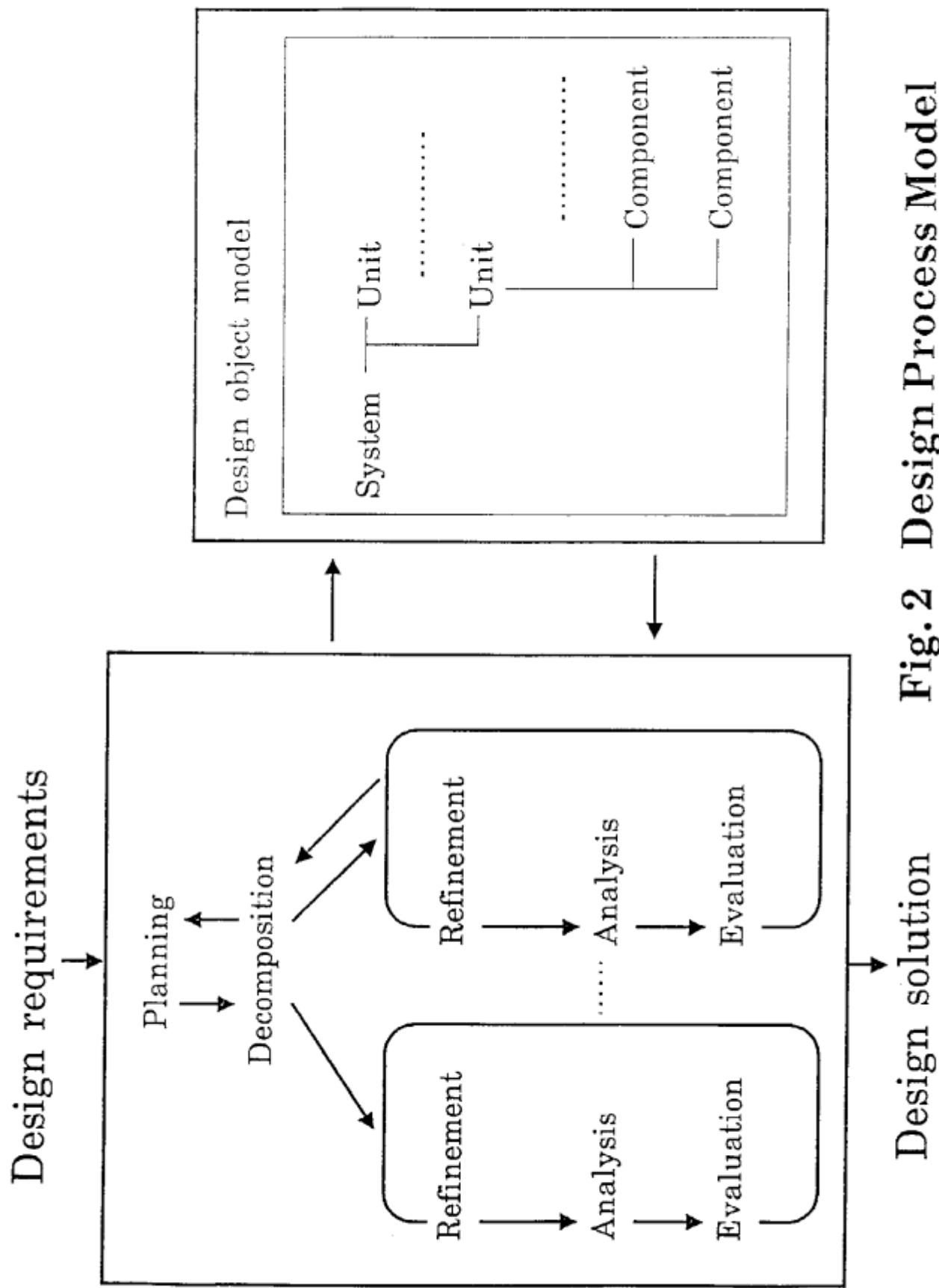


Fig. 2 Design Process Model

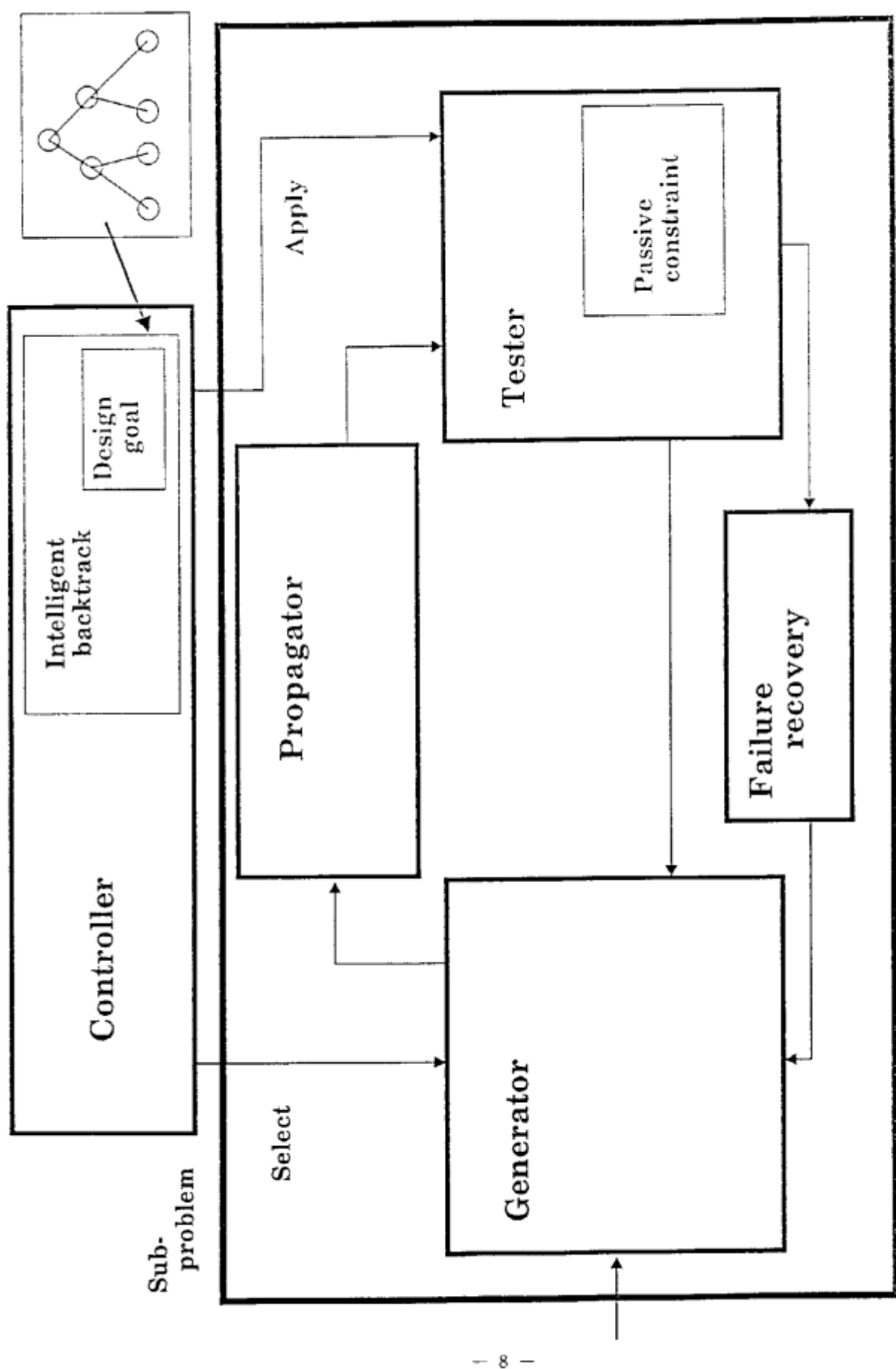


Fig. 3 Problem Solving Model for Constraint Reasoning

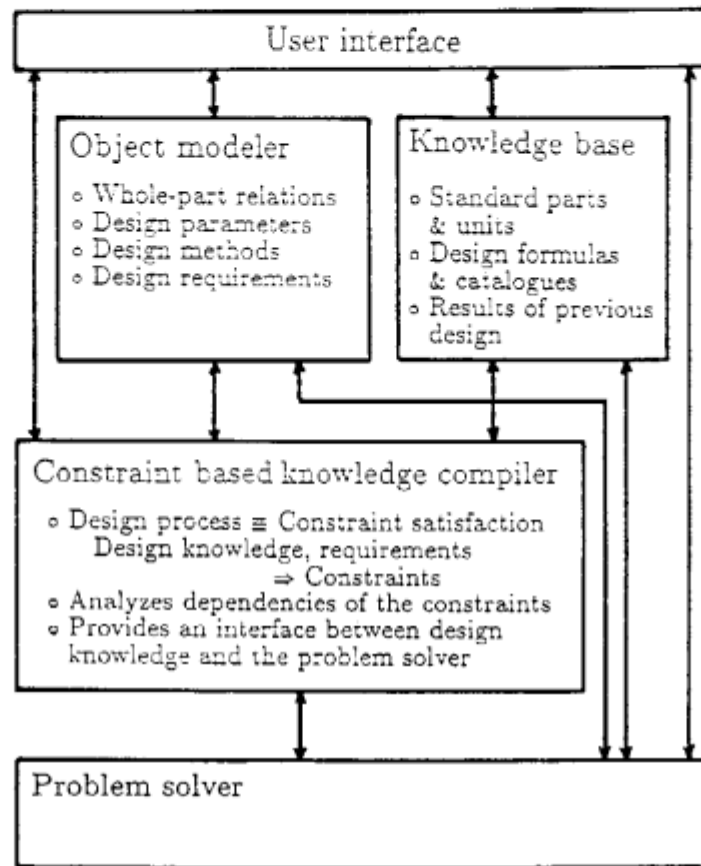


Fig.4 Tool architecture