OUKA
Osaka University Knowledge Archive

Title	機能設計における設計対象のモデリングと形状モデルとの融合
Author(s)	藤田, 喜久雄; 赤木, 新介
Citation	日本機械学会論文集 C編. 1991, 57(535), p. 1058-1065
Version Type	VoR
URL	https://hdl.handle.net/11094/3456
rights	
Note	

Osaka University Knowledge Archive : OUKA

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

Osaka University

日本機械学会論文集 (C 編)
57 巻 535 号 (1991-3)

論文 No. 90-0893A

機能設計における設計対象のモデリングと形状モデルとの融合*

藤田 喜久雄†, 赤木 新介†

Function-based Geometric Modeling Approach for Preliminary Design

Kikuo FUJITA and Shinsuke AKAGI

A function-based geometric modeling approach for preliminary design is developed using an object-oriented programming technique. Preliminary design of engineering systems such as ships and aircraft is a process finding a design object which satisfies several required functions. In the process, it is important to support designers so as to manipulate design objects flexibly. Moreover, it is necessary to manipulate geometries of design entities in order to evaluate their functions. In this paper, we propose a modeling method of these design objects and their geometry. The system cooperates with both the modeling subsystem of design entities and design parameters, and the geometric modeling subsystem. In the system, each modeling element is represented as an object using an object-oriented programming technique. The procedures for modeling elements are carried out by the methods defined in classes in an autonomous fashion, and geometries are automatically generated with the procedures which are defined in objects of design entities. This approach is effective and helpful in representing design knowledge and supporting the designer's manipulation of design objects. Finally, the prototype system implemented with the method is applied to the preliminary design of a ship's hull arrangement to confirm its validity and effectiveness.

Key Words : Design Engineering, Geometric Modelling, Knowledge Representation, Object-Oriented Programming, Preliminary Design, Ship Design

1 緒 言

船舶や航空機などの基本設計[(1)(2)]に代表される各種の機能設計は，設計要求を満足するような設計対象の要目や形状，配置構成などを定めていく過程である．その過程においては，与えられた要求機能に対して設計要目の値を設定しては，それに対応した対象の形状や配置構成を仮定し，設計解の有する機能についてシミュレーションにより検討が行なわれ，その上で，先に仮定した設計要目に修正を加えていき，このような過程の繰り返しによって各種の機能を総合的に満足するような設計解が求められる．しかし，機能設計において検討すべき項目は多岐にわたり，それらをバランス良く満足する設計解を得ることは，経験を積んだ設計者といえども非常に困難な問題となっている．著者らは，このような設計問題に対して，なかでも各種設計要目の決定を行なうパラメトリック設計を取り上げ，オブジェクト指向の概念を導入した設計支援システムを構築する[(3)(4)]とともに，船舶の基本設計に適用して[(5)]，その有効性を検証してきた．しかし，機能設計の本質は設計対象の形状や配置構成を取り扱うところにひとつの特徴があり，パラメトリック設計のみならず，より総合的な立場から機能設計を支援する必要がある．

本研究では，上記のような機能設計における問題点を踏まえた上で，先に著者らが構築した設計支援システム[(3)(4)]をさらに進めて，機能設計における設計対象物の形状や配置構成をも含めた総合的な設計支援を行なうための設計対象のモデリング手法を提案し，支援システムを構築する．すなわち，機能設計における設計対象の操作に着目した上で，設計対象物の実体を個々にオブジェクトとしてモジュール化して表現する一方，設計実体の形状についてはオブジェクト指向のもとで構築したソリッドモデリングシステム上で表現し，上記の設計対象を表現したオブジェクトの属性として扱うようにする．さらに，それを基本として，設計対象の各種パラメータの取り扱いや対象形状の操作を融合した機能設計のための支援システムを構築する．最後に，構築したシステムを船舶設計における区画配置設計に適用して，その有効性を検証する．

*原稿受付　平成 2 年 7 月 4 日.
†正員，大阪大学工学部，(〒 565 吹田市山田丘 2-1).

2 機能設計と設計対象のモデリング

2.1 機能設計の特徴と形状処理 緒言でも述べたように，船舶や航空機などの基本設計[(1)(2)]に代表される各種の機能設計では，設計要求を満足するような設計対象の要目や形状，配置構成などを決定する．このような設計問題の特徴は，各種の要求機能 (性能値) が設計対象物の形状や配置構成によって定まり，さらに，それらの形状は各種の要目に従ってその概略が規定されるという関係にある．例えば，4章で事例として取り上げる船舶の区画配置設計[(6)]は，船舶の内部を様々の船倉に分割していく設計過程であり，以下のようにして設計が行なわれる．すなわち，船体設計によって与えられる船体の形状に対して貨物倉の数や各種隔壁の位置を決定し，それによって規定される各種のタンクや貨物倉の容積や重心位置に基づいて，各載荷状態における安定性やトリムなどの評価項目に対する検討を行ない，その結果に応じて先に決定した区画数や隔壁の位置などを修正していくことにより，設計が行なわれる．このような設計の過程を有効に支援するためには，各種の設計要目から規定される設計対象のシミュレーション的な操作が行なえることが不可欠であり，そのためには，設計操作の対象となる設計実体の形状を柔軟に操作でき，かつ，それによって規定される各種の評価項目の値がシミュレーション的に算出できるようにすることが必要である．

一方，機械設計における形状の取り扱いについては，古くから研究が進められてきており[(7)]，今日では，各種の商用形状モデリングシステムも多方面で実際の設計問題に適用されてきている．しかし，これらのシステムや従来の形状処理に関する研究の多くは，形状そのもののモデル表現や各種解析技術の適用を専ら対象としており，設計における設計対象の操作という点からは，十分な成果を上げるには至っていない．また，形状に付随する各種の情報をフィチャーとして表現することにより設計過程を支援しようとする試みも見受けられる[(8)(9)]が，そのような試みの多くは専ら詳細設計や生産設計を対象としており，上記のような機能設計の過程を十分に支援するには至っていない．

2.2 機能設計支援システムの構成概念 上記のような問題点に対して，機能設計の過程を有効に支援するためには，設計対象における各種の設計知識と設計対象物の形状のモデリングとを有機的に統合することが必要である．図1は，本研究で提案するシステムの基本的な構成を示したものである．すなわち，設計分野のみならず，情報処理の各分野で広く適用されつつある「オブジェクト指向プログラミング[(10)(11)]」のもとで，著者らが先にパラメトリック設計を対象に構築した設計支援システム[(3)(4)]を，設計対象における各種の実体や形状情報をも取り扱えるように拡張する一方，同様の環境のもとで形状モデリングシステムを構築し，両者の協調した機能により機能設計の過程を支援するようにする．

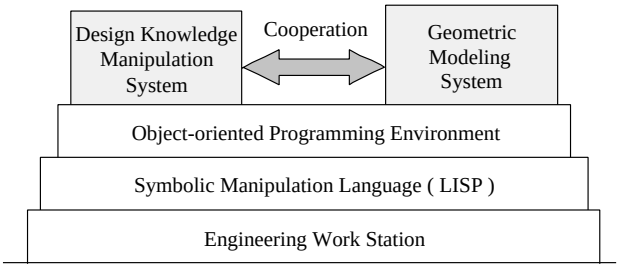


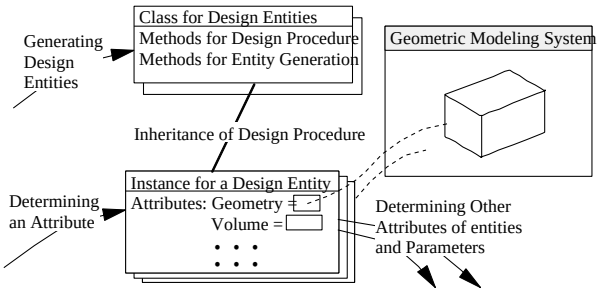
図1 システム融合の基本形態

2.3 オブジェクト指向による設計対象のモデリング 前節で示した融合を行なうためには，機能設計における設計対象の表現方法がその中心となる．機能設計における設計対象の操作は，前節で示したように各種の設計要目の変更を通して設計対象の形状をはじめとする各種の属性を互いに関連させて操作し，その機能シミュレーションを繰り返すことにより，行なわれる．さらに，このような設計対象の操作は，各種のパラメータの値を決定したり，修正したりすることや，それに基づいて設計実体の像を生成したり，その属性を決定・修正したりすることに対応する．そこで，本手法では，そのような設計対象の操作やモデリングを実現する方法として“オブジェクト指向”の概念に基づいた手法を採用する．すなわち，各種設計実体の生成・消去をオブジェクト (インスタンス) の生成・消去に，また，各種属性の決定や修正をオブジェクトの保持する変数の決定や修正に対応付けることにより，設計対象の操作を行なうようにする．さらに，そのような設計対象の操作方法や設計実体の生成方法は，それらオブジェクトのクラスに記述するようにして，オブジェクト指向におけるスーパークラス／サブクラスやクラス／インスタンスの階層的な関係により設計対象に関する設計知識を簡潔に表現できるようにする．図2は，以上のような手法による設計対象の表現を示したものであり，下記の3種類のオブジェクトにより設計対象を表現する．

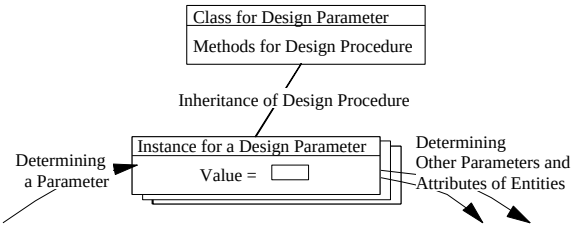
(1) クラスオブジェクト：以下の各種設計対象オブジェクトの属性や設計パラメータの値の操作方法，また，設計実体インスタンスの生成方法を保

表 1　設計対象を操作するためのメソッドとその機能

	設計実体のクラス	設計実体のインスタンス	設計パラメータのインスタンス
decide	自身の保持している実体の生成手続きに従って，設計実体のインスタンスを生成する．	指定された自身の属性の値を自身あるいはクラスが保持する決定手続きに従って決定し，保持する．	自身が保持している決定手続きに従って，対応するパラメータの値を決定し，保持する．
delete	自身の各インスタンスの属性値をメソッド ‘delete’ により消去した後，自身のすべてのインスタンスを消去する．	指定された自身の属性値を消去した上で，それに依存して決定された各種の値や実体のインスタンスをメソッド ‘delete’ により消去する．	自身のパラメータの値を消去した上で，それに依存して決定された各種の値や実体のインスタンスをメソッド ‘delete’ により消去する．
query	自身のインスタンスを参照する．	自身の指定された属性の値を参照する．	自身の保持するパラメータの値を参照する．



(a) 設計実体のモデリング



(b) 設計パラメータのモデリング

図 2　設計対象のモデリング要素

持するクラスオブジェクト．

(2) 設計実体を表現するインスタンスオブジェクト：設計対象物の具体的な実体に対応するインスタンスオブジェクトであり，その属性として実体の形状や各種のマスプロパティーなどを保持する．

(3) 設計パラメータを表現するインスタンスオブジェクト：各種の設計要目や性能値などを決定・保持するインスタンスオブジェクト．

以上のような設計対象のモデリングにより，設計対象やそれに関する知識はオブジェクトとして個々にモジュール化して記述することができる．

さらに，これに対して，上記の設計対象オブジェクトを操作するための基本的なメソッドとして，以下のものをシステムに用意するクラスオブジェクトに定義する．

‘decide’： 設計項目の内容を決定する．
‘delete’： 設計項目の内容を消去して修正する．
‘query’： 設計項目の内容を問い合わせる．

以上の各メソッドの具体的な処理機能の内容を表 1 に示す．表のようにそれぞれの内容は個々に異なったものとなるが，各設計処理を実行するためのメソッドの名称を上記のようにシステム全体で統一しておくことにより，オブジェクト指向における情報隠蔽の性質を生かした支援システムの構築が可能となり，また，各種設計知識の表現におけるモジュール性をさらに高いものとすることができる．

一方，上記の (2) の設計実体オブジェクトにおける形状属性については，形状モデリングシステムにおけるソリッドなどのオブジェクトの名称を設計実体インスタンスの変数として保持・管理するようにする．これを基本として，前出図 1 のように，オブジェクト指向プログラミング環境のもとで構築した形状モデリングシステムを設計知識処理システムに対して協調させ，形状に関する処理は各種設計知識の処理とは独立した形で行なうようにし，かつ，両者の統合により機能設計の過程を支援するようにする．

なお，具体的な処理の方法や設計対象知識の表現方法については次章で具体的に示すことにする．

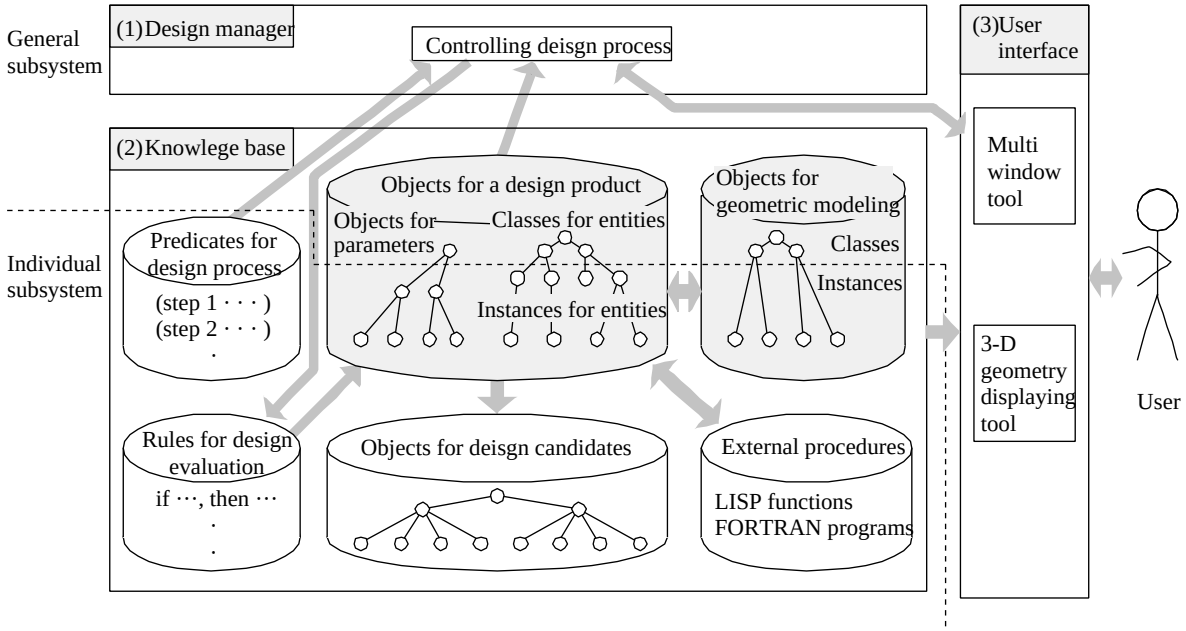


図 3　機能設計支援システムの構成

3　機能設計支援システム

3.1　設計支援システムの構成　前章で示した設計対象のモデリング手法を基本とする設計支援システムの構成を図 3に示す．これは，前出の図 1に対する具体的なシステムの構成を示したものであり，システムは， Lisp 言語により記述したオブジェクト指向プログラミング環境のもとで構築されており，大きく分けて設計マネージャー，設計知識ベース，ユーザーインターフェースの 3 つの部分から構成されている．設計マネージャーの部分は設計における各種の処理を統括する部分である．また，知識ベースの部分は， 2章で述べた各種の設計対象に関するオブジェクトを集めた設計対象知識ベース，形状モデリングのためのオブジェクトデータベース，設計プロセスを記述した述語[3]，設計結果の評価診断を行なうためのルール型知識[3]，設計候補を保持したオブジェクト指向型データベース[12]，などから構成されている．さらに，本システムは，詳細な設計処理を記述した Lisp 関数，図形処理や数値計算との協調をはかるための外部プログラムを呼び出すための機能を備えている[3]．以上のような構成において，図中の一点鎖線の上側の部分は，各種の設計問題に対して共通的に適用できる部分であり，下側は，個別の設計問題における固有の設計知識などを記述する部分となっており，個別の問題に対しては，下側の固有知識ベースのみを記述することにより適用できるようになっている．

3.2　設計知識の表現　次に，具体的な設計問題における設計知識の記述例を図 4および図 5に示す．これらは，後出の船舶の区画配置設計における設計対象知識の記述例である．図 4中 (a) 〜 (c) は設計実体のクラス， (d) はそれらのインスタンスの記述例である．(a) は，船舶の区画に共通するクラスであり，共通的な属性として容積 (capacity) や重心位置 (oxg, kg) の決定方法をインスタンス変数 (iv) のなかに保持している． (b) は， (a) のサブクラスであり，各種区画の中でも貨物倉とトップサイドタンクに共通する属性の決定方法を保持している． (c) は，貨物倉の実体に対するクラスの記述であり， (b) のサブクラスで，貨物倉の実体を生成するための手続きをクラス変数 (cv) に保持している． (d) は， (c) のインスタンスとして設計過程で生成された実体を表現したインスタンスであり，(b) の記述に従って生成された形状オブジェクトの名称 (object4903) や (a) の記述に従って計算されたマスプロパティの値などを保持している．一方，図 5は設計パラメータのインスタンスの記述例である．それぞれのインスタンスは各パラメータの決定方法をインスタンス変数の ‘procedure’ に保持しており，例えば，貨物倉の数に関する記述である (a) には，船体の長さに従ってその数を決定するための手続きが記述されている．

3.3　形状モデリング手法との融合　続いて，本システムにおける形状モデリングとの融合の方法について示す．その基本的な考え方は，前出の図 2に示した通りである．また，形状のモデリング手法について

```
(class (name tank-class)
  (super design-object)
  (iv (capacity nil (procedure ( integral 1   )) )
      (oxg nil (procedure
                 (- integral 'x / my capacity)) )
      (kg  nil (procedure
                 (  integral 'z / my capacity)) )) )
```

(a) 船体の区画のクラス

```
(class (name cargo&tst-class)
  (super tank-class)
  (iv (parent-tank  nil)
      (pre-bulkhead nil (procedure 'dummy-bulkhead) )
      (aft-bulkhead nil (procedure 'dummy-bulkhead) )
      (geometry nil (proceudre
        (let (( tank
                 (geometry of my parent-tank)  )
              ( pre-bulkhead
                (geometry of my pre-bulkhead) )
              ( atf-bulkhead
                (geometry of my aft-bulkhead) ))
         (if (not (eq pre-bulkhead 'dummy))
             (setq tank (send tank
                              'get-lower-side-object
                              pre-bulkhead) ))
         (if (not (eq aft-bulkhead 'dummy))
             (setq tank (send tank
                              'get-upper-side-object
                              aft-bulkhead) ))
         tank )    ) )) )
```

(b) 貨物倉とトップサイドタンクに共通のクラス

```
(class (name cargo-holds)
  (super cargo&tst-class)
  (cv (procdure
       (let (( n number-of-bulkheads ))
        (do (( i 1 (1+ i) )) ((> i n))
         (let (( inst
                  (implode$ i      '-cargo-tank    ) )
                ( pre
                  (implode$ (1- i) '-cargo-bulkhead) )
                ( aft
                  (implode$ i      '-cargo-bulkhead) ))
          (send !self 'create-instance inst)
          (if (not (= i 1))
              (send inst 'put-value 'iv
                         'pre-bulkhead pre))
          (if (not (= i n))
              (send inst 'put-value 'iv
                         'aft-bulkhead aft))  )) ) ))
  (iv (parent-tank nil
                   (procedure 'whole-cargo-tank) )) )
```

(c) 貨物倉のクラス

```
(instance (name 3-cargo-tank)
  (class cargo-holds)
  (iv (pre-bulkhead 2-cargo-bulkhead)
      (aft-bulkhead 3-cargo-bulkhead)
      (parent-tank  whole-cargo-tank)
      (geometry     object4903      )
      (capacity     7061.042        )
      (oxg          -28.52644       )
      (kg           8.319828        )) )
```

(d) ある貨物倉のインスタンス

図 4　設計実体オブジェクトの記述例

は，各種のものが提案されている[7]が，事例として取り上げる船舶の基本配置設計における形状の取り扱いが容易であることなどから，B-Reps による表現を用いることにする．なお，本手法における具体的な形状モデリングの手法は，必ずしも B-Reps である必要は

```
(instance (name number-of-bulkheads)
  (class set-up-parameter)
  (iv (procedure
        (( l < 170. ) --> 5
         ( l < 180. ) --> 6
         ( l < 230. ) --> 7
         ( l > 230. ) --> 9 ) )) )
```

(a) 貨物倉の数

```
(instance (name light-weight)
  (class definitive-parameter)
  (iv (procedure ( ws + wo + wm ) )) )
```

(b) 軽荷重量

```
(instance (name le) (class set-up-parameter)
  (iv (procedure
        ( 1.1 * length of 'main-engine + 16.5 ) )) )
```

(c) 機関室の長さ

図 5　設計パラメータインスタンスの記述例

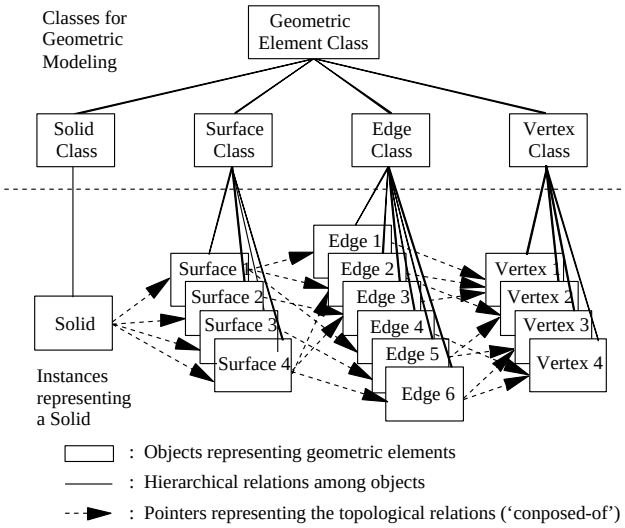


図 6　オブジェクト指向による形状のモデリング

なく，図 1あるいは図 2に示したように，設計対象に関する処理を行なう部分と形状処理の部分が完全に独立しているため，例えば，形状処理の部分を CSG によるシステムに置き換えることによって，CSG による処理との融合を行なうことも可能である．

図 6に，オブジェクト指向による形状表現のデータ構造を示す．本システムでは，B-Reps による表現のために用いる立体，面，稜線，頂点などの形状要素をそれぞれオブジェクト (インスタンス) として表現し，それらの間の連結関係をオブジェクト間の関係として記述するようにする．これにより，形状に関する各種の処理も前章で示した設計知識の操作と同様，個々のオブジェクトのメソッドとして記述することができ，

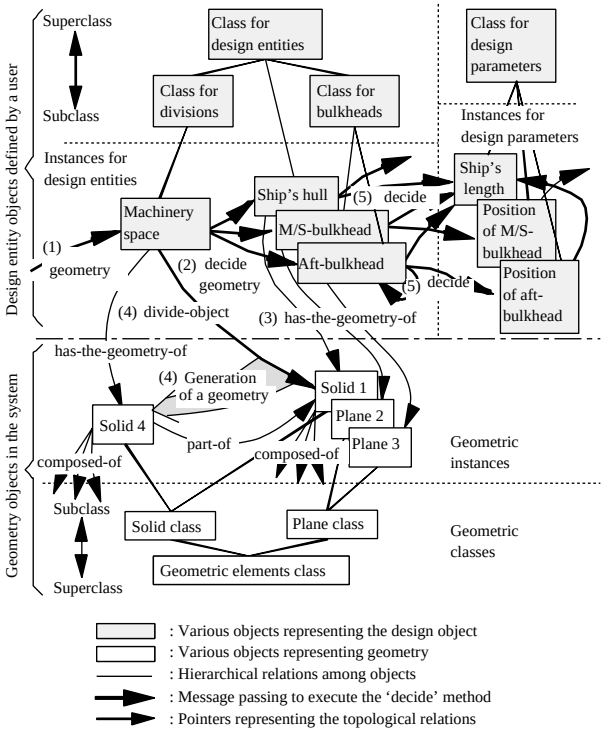


図 7　設計処理の方法

処理の記述も容易に行なえるようになる．例えば，体積や重心位置などのマスプロパティの計算は，対象形状を一様な多面体に限定した場合，ガウスの定理やグリーンの定理を用いることによって，立体の境界面を構成する多角形の境界線分上の線積分により求めることができ，複雑な積分計算は各種形状要素に定義されたメソッドにより個々に分散させて実行することができる．

3.4　設計処理の方法　以上のような設計知識の表現や設計実体の形状モデリング手法による設計処理の方法を，後出の船舶の区画配置設計における例をもとにして，図 7に示す．

図は，船舶の機関室区画の形状を決定するための処理過程を示している．2.3節でも示したように，このような処理はすべてシステムに用意された基本的なメソッド (表 1) をメッセージによって起動していくことにより行なわれる．まず，その処理を行なうために，‘機関室 (Machinery space)’ のインスタンスに対して形状の決定を求めるメッセージが送信される (①). メッセージを受けたオブジェクトは，自身が保持したりクラスから継承されているインスタンス変数の記述に従って，形状を決定するために必要となる ‘船体 (Ship’s hull)’ や ‘機関室隔壁 (M/S-bulkhead)’ などの設計実体の要素を用いて形状の決定を行なう (②). その過程において，それらの設計実体の形状がシステム内部で形状インスタンス ‘Solid 1’, ‘Plane 2’, ‘Plane 3’ として定義されると，各設計実体に関するインスタンスは，それぞれの形状インスタンスとインスタンス間の関係 ‘has-the-geometry-of’ により連結される (③). そして，機関室の形状を決定するための実際の処理は，それらの形状インスタンスの間で行なわれ，機関室の形状が ‘Solid 4’ として決定される (④). そのほか，以上のような処理の過程で必要となる ‘船体の長さ (Ship’s length)’ , ‘機関室隔壁の位置 (Position of M/S-bulkhead)’ , ‘船尾隔壁の位置 (Position of aft-bulkhead)’ などの各種の設計パラメータに対しても，それぞれのインスタンスに対してメソッド ‘decide’ を起動するメッセージが送信され，各設計項目の決定が行なわれる (⑤).

以上のような処理に対して，形状に関する各種の特性は，設計実体のインスタンスが保持する形状オブジェクトの名称を通じて算出されるようになり，機能設計における性能の評価をシミュレーション的に実行することができる．

さらに，設計修正に関しても，上記の処理過程における各設計項目を決定するための ‘decide’ を起動するメッセージの送信をシステムの内部で記録し，設計項目間のネットワーク上の依存関係を管理することにより，ある設計項目の内容を修正した場合に，それに基づいて決定された設計項目の内容を再帰的に修正することも可能になる．例えば，機関室隔壁の位置が変更された場合には，機関室隔壁や船体の設計実体の形状が変更され，その具体的な形状である ‘Plane 2’ や ‘Solid 4’ などの形状オブジェクトが一度消去されて，変更された設計項目の内容に従って，新たに形状オブジェクトが生成される。これによって，各種の機能もそのような変更された形状のもとで新たに算出されるようになる．

以上のような設計処理の実現方法により，設計対象に対する操作と具体的な形状に関する操作が完全に分離され，前出の図 4に示したようにして単に設計対象オブジェクトに形状の定義方法を記述するのみで，機能設計における形状処理が実行できるようになる．また，設計対象や形状に関する各種の知識やデータがそれぞれ個々にオブジェクトとして表現されるため，オブジェクト指向における「情報隠蔽の性質」をシステムの構築や設計知識の記述に生かすことができ，各種の処理が自律的に行なわれるようになる．

なお，本システムは，エンジニアリングワークステーション Sun SPARC Station 1 上で Common Lisp

を用いて記述し，一部ユーザーインターフェースの機能は SunView 1 や SunCore を用いて C 言語 により記述した．

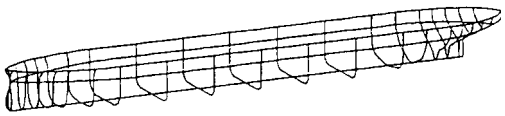
4　事例 — 船舶の区画配置設計への適用 —

4.1　船舶の区画配置設計の過程　船舶の区画配置設計[6]は，2.1節でも述べたように，船体の長さや幅などの主要目の値に従って決定された船体の内部を各種の隔壁によって，貨物倉やバラストタンクなどに分割して，船体の区画を決定する過程である．その過程では，各種設計項目の設定に従って区画を分割した上で，シミュレーションにより，安定性やトリムなどの要求機能を満足するかどうかを検討しながら設計が進められる．これらの機能は分割された区画の容積や重心位置により定まるため，区画配置設計は，設計対象物の形状の操作を介して設計項目の設定と機能の評価を繰り返して行なう典型的な機能設計の問題となる．

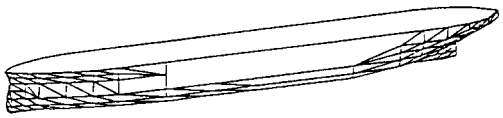
4.2　適用事例　図 8に本システムによるあるばら積み貨物船の区画配置設計の実行例を示す．図中 (a) は，船体設計により与えられた船体形状のワイヤーフレーム表現であり，まずこれを以下の処理のために，(b) のような B-Reps によるソリッド表現に変換する．続いて，図中 (c) のように，船体を船首バラストタンク・トップサイドタンクの部分・貨物倉の部分・二重底タンクの部分・機関室・船尾バラストタンクの各部分に分割し，さらに，船体中央部の貨物倉等の部分を詳細に分割する．その上で，トリムや横安定性 (GM 値) などをいくつかの載荷状態に対して求め，それをシステムの有する設計評価機能[3]により診断した結果が (d) である．この場合，いずれの載荷状態 (グレン満載時・ノーマルバラスト時) においても，トリムが大きすぎることが指摘されている．これに対して，設計者が，機関室の位置を 5m 前方に移動させるという設計変更を行なっている．図中 (e) は，その設計変更を示したものである．このような設計変更に対して，再び機能シミュレーションを行ない，その結果を診断した結果が (f) であり，上記の問題点のうちの一つが解消されている．以上のように，本システムの有する機能により、設計変更と機能シミュレーションを繰り返しながら、設計を進めることができる．

5　結　言

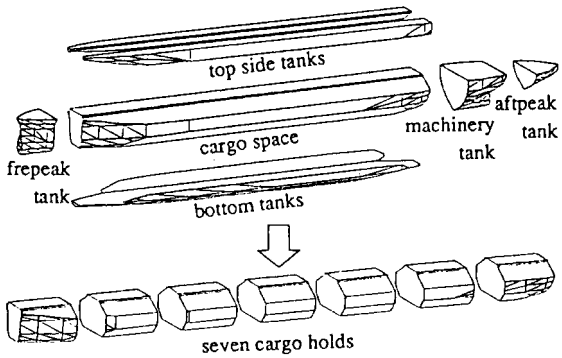
本研究では，オブジェクト指向の概念に基づいた機能設計における設計対象のモデリング手法を提案し，それに基づいた設計支援システムを構築した．機能設計では，設計対象物の実体やその形状，また，実体の



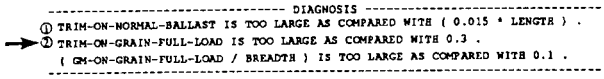
(a) 船体のワイヤフレーム表現



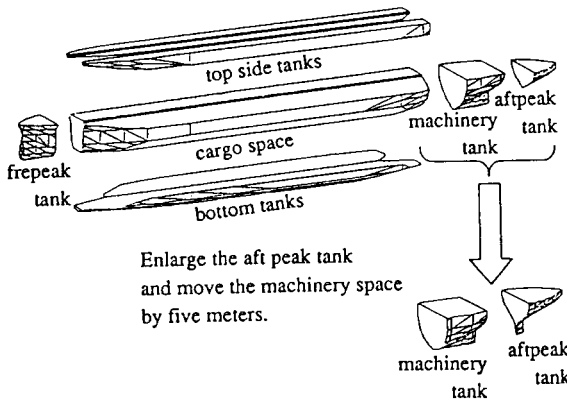
(b) 船体のソリッドモデル表現



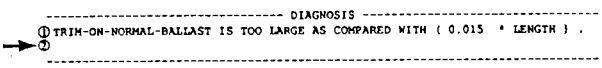
(c) 船体の分割



(d) 設計評価機能による診断の結果



(e) 機関室の位置の変更



(f) 設計評価機能による診断の結果

図 8　船舶の区画配置設計における設計処理の例

有する機能などの各種の項目を柔軟に操作しながら設計を進めていく必要がある．本手法においては，設計対象の実体や，各種設計項目や評価項目などのパラメータをそれぞれモジュール化してオブジェクトとして表現する一方，設計対象物の形状をオブジェクト指向のもとで構築したソリッドモデリングシステムにより表現することにより，機能設計に係わる設計対象を自律分散的に表現して，それらの間のネットワーク上の関係をシステムにおいて管理することにより，設計者による柔軟な設計操作を支援することができるようになった．また，構築したシステムを事例として船舶の区画配置設計に適用し，その有効性を検証した．

なお，本システムに対する将来の拡張として，今後，より広範な設計問題への適用を可能にするために，CSG による形状モデリングや各種の形状属性の記述を導入し，システムの機能を強化していく予定である．

文　献

(1) 冨田, 船舶基本設計論, (1982), 丸善出版サービスセンター.
(2) Torenbeek, E., *Synthesis of subsonic airplane design*, (1988), Delft University Press.
(3) 赤木・藤田, 機論, **54**-500, C (1988), 1017.
(4) 赤木・藤田, 機論, **54**-505, C (1988), 2300.
(5) 赤木・藤田, 関西造船協会誌, 206, (1987), 65.
(6) 教育テキスト研究会, 商船設計の基礎知識 (上巻), (1977), リプロ.
(7) 沖野, 自動設計の方法論, (1982), 養賢堂.
(8) Drake, S. and Sela, S., *Mechanical Engineering*, **111**-1, (1989), ASME, 63.
(9) 今村 ほか, 第５回設計自動化工学講演会講論集, (1989-7), 58.
(10) Cox, B. J., *Object-Oriented Programming: An Evolutionary Approach*, (1986), Addison-Wesley.
(11) 大特集：オブジェクト指向プログラミング，情報処理, **29**-4, (1988-4), 289.
(12) 赤木・藤田・窪西, 機論, **54**-497, C (1988), 228.