ICOT Technical Report: TR-404

TR-404

オブジェクト指向プログラミングにおける構成支援方法

片山　佳則(富士通)

June, 1988

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

オブジェクト指向プログラミングにおける構成支援方法

A Supporting Method for Structuring Objects in Object-Oriented Programming

片山　佳則

富士通㈱　国際情報社会科学研究所

概要

本論文では，オブジェクト指向プログラミングにおいて問題となる，オブジェクトの構成や内部記述を決めるための支援方法を提案する．具体的には，対象モデルからオブジェクト指向の特徴であるメッセージ処理とデータの制御などの操作を構造的情報として取り出し，それらを分析することで対象に適したオブジェクト構成や記述を指摘する．この構成支援機能を実現することにより，開発者はオブジェクト構成にわずらわされず，表現したい内容についての検討が十分にできるようになる．また，オブジェクト構成を実現するための関係表現の概念がシステムで統一されるため，改良・拡張・再利用が容易になる．さらに本方法は，オブジェクト表現のスタイルを規定するための一方法にもなる．

1．はじめに

オブジェクト指向概念を導入した表現システム／言語は，Smalltalk ［6］やFlavors［15］［18］，Loops ［2］などに始まり，様々なものが作成され検討されている［1］［4］［5］［14］［17］［20］．これらの表現システム／言語は，階層によるモジュラリティ，プログラム開発・拡張の容易性，再利用に関する機能など様々な利点を持っている．しかし，これらの利点を十分に活かしたオブジェクト構成や内部の記述を行うことは困難である．一般には，その技法に慣れた『エキスパート』に任されている．しかし，そのエキスパートでも簡単に実現できない．特に新しい分野を対象として知識等を表現する場合には，その対象に対して最適なオブジェクトの関係や各オブジェクトの機能を決めることは困難である．これらを容易に実現するために，対象に適したオブジェクト構成等を指摘できる機能を，表現システムが持つことが必要になる．

オブジェクトの関係や各オブジェクトの機能を最適に実現するには，記述量や読み易さ等様々な要因を検討しなければならない．これらの最適性は，最終的には再利用の可能性を導くことである．本論文では，表現システムがオブジェクト表現のために備えている関係表現を，統一した意味でできる限り利用して，対象を表現することを最適化と考える．この最適化は，その関係表現が持つ機能から，記述量の効率（空間的効率）を導き，関係表現の利用が統一した意味で行われることで，読み易く，オブジェクトとして実現した機能を明確にする．このようにして，最適化は，最終的に再利用し易い表現を実現する．

オブジェクト指向の利点として，モデリングが容易であることが挙げられる．この点に視点を置き，表現したい知識やそのアルゴリズムが明解で，目的とするオブジェクト構成が明らかな場合，オブジェクト指向を導入した表現システム上に，どのような手順で開発すべきかの支援として，段階的開発支援［10］を検討した．また，表現されたオブジェクト等を検索・管理するための支援［16］［19］なども検討されている．しかしながら，この段階的支援の検討や検索等においても，設定したオブジェクト構成自体が重要であり，各オブジェクトの機能とそれらの関係表現をどのように規定するか，が大きな問題となる．

したがって，再利用を意識し，対象をどのようにオブジェクトに対応させるかの判断やそのオブジェクト間の関係表現を最適に実現するための機能が必要となる．本論文では，

これらの機能を実現する方法とその支援機能の適用事例を示し，具体的な機能・有効性を考察する．

次章では，構成支援の対象とする関係表現とその意味を明確にする．第３章では，対象とする関係表現についてオブジェクト構成の支援方法を論じる．第４章において，第３章で示した支援の適用事例を示し，最後に考察とまとめを行う．

２．オブジェクトの関係表現

オブジェクト指向概念を導入したシステム／言語が持つオブジェクトの関係表現は，表現システムとして特徴的なものが多数存在する．しかし，下記のような，名称は異なるが共通な機能を実現しているものもある．

(1)包含関係を表すもの：

これは，集合間の関係として部分集合関係(Subset/Superset) を表し，知識表現におけるIS-A関係や種類についての関係であるAKO(A Kind Of)関係に対応する．述語間の関係としては，一般化と条件づけによる特殊化(Generalization/Specialization) を表現する．さらに細かい分析［３］も可能である．この関係では，属性・メソッド等の継承が行われることが大きな特徴である．この関係によりインヘリタンス機能を実現し，階層によるモジュール性や概念間の関係を記述する．

(2)部品関係を表すもの：

これは，知識表現でのPART-OF(HAS-A)関係に対応し，部分集合としての見方ではなく，機能・内部構造に関して部分関係を表すものである．部分は，全体から利用されるものである．基本的には，構成要素としての関係が強い［13］．この関係では，属性・メソッド等を継承して利用する機能はない．クラス間の関係が対応するインスタンスに対してもそのまま受け継がれ，インスタンス間の関係を表す．

この他にも，オブジェクト間の関係表現としてさまざまな関係が考えられている．しかし，対象とする知識やモデルを表現するために必要な基本的関係は，この２つである．したがって，本論文で対象とするオブジェクト表現の支援機能は，この２つの関係を重視して検討する．以後(1)をsuperset関係，(2)をsubparts関係と呼ぶ．これらの関係表現によるオブジェクト構成を，Fig．１に示す．

Fig．１挿入＞

これらの関係とメッセージ交換との係わりを，次に示す．

各オブジェクトは，処理可能なメッセージを他から自由に受け取れる．オブジェクトのメッセージ交換は，送信側から見れば，情報を渡して計算（処理）を進めてもらうことである．したがって，機能が明確である対象を表現しているオブジェクト間のメッセージ交換は，送信対象オブジェクトが常に決定している場合や計算結果を渡す等のために送信対象オブジェクトがメッセージ内のデータに含まれている場合を除き，そのメッセージ送信対象がオブジェクト間の関係表現で規定されている．実際のメッセージ送信対象となるオブジェクトは，実行時に決まる．このため，クラスレベルでメッセージ交換関係を表現する場合には，表現のための基本的関係から，superset関係による継承パスをメッセージ送信の形で動的に探索する方法か，subparts関係により内部構造の一部として部品化して表

されているオブジェクトに対してメッセージ送信する方法のどちらかである．このように，メッセージを計算，処理の受け渡しと考えた場合，あるオブジェクトが他のオブジェクトと直接メッセージ交換をするには，そのオブジェクトとsupersetかsubpartsの関係を持たなければならない．

subparts関係によるメッセージ交換の考え方は，Actor 理論におけるacquaintances［8］という通信情報の局所性に対応して考えられる．superset関係によるメッセージ交換は，メッセージを処理する環境を上位クラスに広げる（現在の対象クラス＋上位クラス＋・・・）ものである．この２つの関係を用いて，表現システム上に対象を最適に構築することが，オブジェクト構成支援の目的である．

この支援を実現するためには，各関係表現の機能を効果的に利用できるように，対象の機能を分割する方法を規定しなければならない．対象を機能の視点で分割することは可能である．しかし，実際にどの機能を分割するか，また，その機能をどのオブジェクトにどのように記述すべきかが問題である．この機能の配置方法によってオブジェクト間の関係表現を決められる．

superset関係のためのクラスの分割方法は，部分集合関係では，属性の継承を有効に利用できることに視点を置き，記述量の関係から適切な記述位置を定める必要がある．他方，一般化のためのクラス分割では，記述対象の意味などの要因を考慮する必要がある．この関係は，メッセージ交換の処理関係でなく，クラス間の表現レベルでの関係として決められる．したがって，他のオブジェクトが持つ機能との関係から，記述すべき階層の下限を判定することはできるが，適切な記述位置については表現システム側で規定することは難しい．

subparts関係におけるクラスの分割方法は，継承関係を持たないため，オブジェクト単体としての機能と利用関係に視点を置き，部分関係を的確に分析しなければならない．この関係も，クラス間の関係として示されるが，この関係は，メッセージ交換の処理と深く関係する．したがって，オブジェクトとしてまとめられる機能とその処理関係から，その記述を規定できる．

これらのことを踏まえて，各関係表現に対する支援において，以下では，subparts関係に注目した支援機能を論じる．再利用を考慮する場合，この関係によって記述される部分オブジェクトが，機能をコンパクトにまとめたオブジェクトとなるため，再利用の対象として適当なものとなる．また，まとまった機能を実現する部分オブジェクトとの関係から，全体オブジェクト自身の機能分析が容易になり，全体の再利用もし易くなる．これらの観点から，subparts関係を支援することは，再利用を促進するための基本的な構成を与えることになる．

3．オブジェクト分割と操作関係による構成の支援

本章では，関係表現におけるsubparts関係に対して，オブジェクトの分割・構成を決める方法を示す．分割は，対象を表現しているクラスが実現している機能の構造的情報を分析することで行う．ここで，対象の機能を抽出するためにセットを考える．クラスの持つ機能の部分的まとまりをこれらのセットで抽象化する．そして，どのセットを分割する対象とし，新たなオブジェクトと定めるかなどを，セット間の操作関係の情報やセット作成時の条件を分析し，関係表現が持つ意味に対応させて決定する．

3.1　単位セットに基づくオブジェクトの分割

オブジェクトを分割する場合，オブジェクトが備えている機能のセマンティクスを分析する方法とシンタックスを分析する方法が考えられる．前者の分析結果は，概念的関係に深く結び付き，superset関係に関する支援に利用できる．ただし，このセマンティクスを分析するための知識処理機構が問題になる．後者は，オブジェクトの構造的観点での分析であり，subparts関係の情報として有益なオブジェクトの持つ機能に対する最小単位等が得られる．この最小単位の情報と，subparts関係が持つ意味を融合させることにより，subparts関係を考慮した構造レベルでの構成支援が行える．また，この支援の際に得られた構造情報や分析結果は，関係表現全般に利用でき，superset関係を導くための初期情報にもなる．そこで，本論文では，後者の分析方法を検討する．

オブジェクトの分割として，基本になるシンタックス情報は，スロットとメソッドの操作や処理に関するものである．特に，メソッド間のメッセージ交換関係と，メソッドからスロットへの操作情報が，オブジェクトの機能を分析するために重要になる．シンタックス情報の関係状態から，オブジェクト内部のセットを探索することで，オブジェクトの構造を把握し，さらに分割を支援できる．セットが複数得られない場合は，メソッド等の記述の変更や，機能の付加が行われない限り，対象オブジェクトを分割できない．

セットの探索方法は，各スロットや各メソッドについて結合すべきスロットやメソッドを取り込む形式で行う．結合規則はスロット，メソッドそれぞれの視点から次の２つが考えられる．subparts関係の機能を考慮すると，これらの関係で結ばれたメソッドやスロットは分割できない．

(a)　スロットの更新(put)操作を持つメソッドは，そのスロットと結合する．

(b)　メッセージ交換関係を有向グラフに対応させた場合，グラフとして強連結(strongly connected)［７］となるメソッドはすべて結合する．

これらの結合規則で作成されるセットは，オブジェクトが備えた機能の単位を表す．このセットを単位セットと呼ぶ．この単位セットがオブジェクトとして分割できるかどうかによって，subparts関係の構成支援が行える．オブジェクトとして独立できない場合でも，どの程度基本的な機能になっているかによって，superset関係の支援情報として利用できる．ここで，基本的な機能であるかどうかの判断は，そのセットが他からメッセージをどれだけ受信するかによって判定される．単位セットはオブジェクト概念としての，機能の最小単位を表している．

対象とするセットを単なる機能でなく，オブジェクトとして分割するためには，まず，それ自身が特有の属性と機能を持つ必要がある．オブジェクト表現では，このような特有の属性と機能をそれぞれスロットとメソッドで表現する．ただし，メソッドで表現される機能は，属性を表しているスロットに関しての操作だけでなく，一つのオブジェクトとして取り上げるだけの機能を実現したものでなければならない．これはsubparts関係を意識した，分割のための基本規則である．

＜分割条件１＞
新しくオブジェクトとして分割するためには，スロットとメソッドを持たなければならない．

単位セットの中で，この分割条件を満足し，オブジェクトとして分割できるセットを最小セットと呼ぶ．結合規則(a)から始めて作成される単位セットが，スロットを持つセットであるから，オブジェクトとして分割できる最小セットと呼べる．結合規則(b)だけで作成される単位セットはスロットを持たないため，オブジェクトとして分割できない（最小セットではない）．ただし，このセットはメソッド間の関係であることから，メソッドセットと呼ぶ．メソッドセットは，メッセージ処理機能の観点で，機能の最小単位であるから，superset関係の支援として利用できる情報である．このようにして，統合規則(a)，(b)から作られる単位セットは，オブジェクトとしての分割の候補と考えられる最小セットと，機能を実現する最小単位であるメソッドセットに分けられる．最小セットは，subparts関係を検討するために直接利用でき，すべての単位セットは，superset関係を検討するための情報となる．

3.2 基本セットに基づくオブジェクトの分割

簡単なオブジェクトを対象にして，スロットを中心にした機能を取り出すには，前節の単位セットレベルでのオブジェクト分割の情報で処理できる．しかし，複雑な機能や多数の機能を備えたオブジェクトでは，結合規則(a)，(b)から作成される単位セットに含まれないメソッドが多数存在する．結合規則(a)により，すべてのスロットは，いずれかのセットに含まれ，メソッドは，結合規則(b)の強連結関係から，機能の核となるような強い結びつきを持つものがまとめられる．しかし，それほど強い結びつきを持たないが，様々な機能を実現するメソッドや，各単位セットと機能的に切り離せないメソッドもある．そこで，セット内に取り込まれないメソッドに対して，取り込むように規則を緩和することで，各セットが持つ機能を充実させることを検討する．

(c) 各単位セット（最小セットやメソッドセット）からメッセージを送信されるメソッドは，そのセットと結合する．

これは，メソッドの結合関係に強連結な関係を意識せず，最小セットやメソッドセットから，さらにメッセージ送信を受けるメソッドやメソッドセットを結合する規則である．この結合規則により，新たにメソッドセットが生まれる場合もある．これまでの結合規則に，この結合規則(c)を加えてセットを検討することで，各セットの機能が充実したものに

なる．また，セットに含まれないメソッド数も減少することから，オブジェクトの機能の判定も容易になる．結合規則(a)(b)(c)の適用により作成されたセットの中で，分割条件１を満足し，オブジェクトとして分割できるセットを基本セットと呼ぶ．この基本セットは，最小セットが新たなメソッドやメソッドセットを取り込むことで作成されるセットである．

結合規則(c)の他にも，各セットがメッセージを受信するメソッドを対象に取り込む規則や，スロットの参照関係のメソッドを対象に取り込む規則を新たな結合規則として検討できる．しかし，これらのメソッドの関係は，subparts関係の関係表現を通して実現できる(3.3節に示す)．したがって，本論文では結合規則として規定していない．

3.3 subparts関係のための操作関係

これまでに示したセットがpartとなるかどうかを分析するためには，セットがオブジェクトとして分割できるだけでなく，全体に対する部分オブジェクトとしての操作関係を満たさなければならない．これらの情報によって，partとなるセットを見つける．そこで本節では，全体オブジェクトと部分オブジェクトが持つスロットとメソッドにおける操作関係を明確にする．これは，２章で論じたsubparts関係の意味に対応して規定される．

＜部分オブジェクトと全体オブジェクトの操作関係＞

(1) 全体オブジェクトから部分オブジェクトが持つスロットやメソッドへの操作関係の規定

1-1 メソッドは自由に呼びだせる．

1-2 スロットの参照は可能．

1-3 スロットの更新はできない．

1-1,2 について，部分オブジェクトであるから，メッセージ送信やスロット参照について，制約を受けない．逆に，部分オブジェクトは，全体オブジェクトから利用される形式（部品としての関係）になっている必要（２章(2)）があるため，メッセージを受け取る形式でなければならない．したがって，1-1 は，次のように置き換えられる．

1-1'メソッド呼び出しをしなければならない．

これは，partとして分割するための基本条件でもある．

＜分割条件２＞

部分オブジェクトとして分割するためには，全体オブジェクトから利用（メッセージ呼び出しを受ける）されなければならない．

この操作関係における制約は，スロットの更新についてのみである．これは，オブジェクト表現としての規定である．特に，1-3 は，表現システムによって，更新についての特別な手続きを用意している場合があるため，システムによって，メッセージ送信と直接操作による違いを限定する必要がある．

(2) 部分オブジェクト側から全体が持つスロットやメソッドへの操作関係の規定

2-1 メソッドは呼び出せない．

2-2 スロットの参照はできない.

2-3 スロットの更新はできない.

部分オブジェクト側から全体に対しては，すべての操作が制約を受ける．これは部分側が主になって全体に対して処理を受け渡す操作を行わないことが基本（2章(2)）となるからである.

2-1 は 1-1に対応し，部分オブジェクトは全体側から利用される形であり，その逆にはなれないことを示している〔メッセージ交換については，返答のためだけの場合を考えられるが，ここでは，情報伝達のためだけにメッセージ交換を行うことを考慮していない〕.
2-2 は 1-2に対応し，部分機能がその処理において全体の持つスロットに対する操作を必要とする場合，それは部分機能である部分オブジェクトとして閉じていないことになる.
2-3は，1-3 と同様，オブジェクト表現としての規定である.

セットが部分オブジェクトとして分割できるかどうかは，ここで示した部分と全体の関係を，各セットについて解析することで判断できる.

3.1 節の規則(a)，(b)で作成された最小セットについて，部分オブジェクトとなるかどうかを分析することで，オブジェクトのsubparts関係の構成支援が行える．さらに，3.2 節の，機能を充実させるための規則(c)で作成される新しい基本セットについて，部分オブジェクトとなるかどうかを分析することで，複雑に実現された機能についても，subparts関係の構成支援が行える.

オブジェクト内の各セットの情報は，それが部分オブジェクトを実現できなくても，オブジェクト内部の構造を把握したり，superset関係の支援情報として十分利用できる.

3.4 オブジェクトの構造情報と関係表現の対応

表現システムがオブジェクトから得られる内部の構造的な情報は，スロットやメソッドの操作関係である．対象オブジェクトが持つスロットに対しては，各スロットがどのメソッドから参照されるか，または，更新されるかの情報である．メソッドに対しては，各メソッドがメッセージ送信するものが何であるか，または，何からメッセージを受信するかであり，さらに，各メソッドは，どのスロットを参照しているか，更新しているかなどの情報である．これらの構造情報について，結合規則(a)～(c)に当てはまらない関係を分割した時点で，分割されて作成されたセット間の関係表現を決定できる．分割した関係が 3.3 節の部分オブジェクトと全体オブジェクトの関係を充たしていれば，subparts関係を支援できる.

以下では，subparts関係の関係表現としての対応を示すが，3.1 節で論じたように，オブジェクトとして分割できるための基本条件（分割条件1）と部分オブジェクトとして分割するための基本条件（分割条件2）は，満足している必要がある.

3.4.1 最小セットに対するsubparts関係表現

これは，結合規則(a)と(b)だけで作成される単位セットを対象にした構成支援である.

〔A　スロットとメソッドの操作関係と関係表現〕

・スロットがメソッドから更新される場合，結合規則(a)より，分割できない．
・スロットがメソッドから参照されているだけの場合，結合規則に当てはまらないことから，分割できる．

関係表現との対応：

＜分割条件3：スロットとメソッドの操作関係について＞
3.3 節のすべての操作関係を満たすことから分割する／しないに関わらず，スロット側のセットはpartとなり得る†1．

［B　メソッド間のメッセージ交換情報と関係表現］
・メソッド間の操作関係が，メッセージ送信／受信のどちらかだけの場合，結合規則(b)に当てはまらないことから，分離できる．
・メソッド間の操作関係が，メッセージ送信・受信の両方である場合，メッセージ交換関係が強連結であるから，結合規則(b)より，分離できない．

関係表現との対応：

＜分割条件4：メソッド間のメッセージ交換関係について＞
分離した場合，3.3 節の操作関係2-1 より，メッセージを送信する側はpartとなり得ない．分離しない場合は，3.3 節のすべての操作関係を満たすことからそのセットはpartとなり得る．

3.4.2 基本セットに対するsubparts関係表現

［A　スロットとメソッドの操作関係と関係表現］
この関係表現は，3.4.1 の場合と同様である．

［B　メソッド間のメッセージ交換情報と関係表現］
メソッド間の操作関係が，メッセージ送信／受信のどちらかだけの場合，結合規則(c)により，送信側がセットである場合は分離できない．つまり，そのメソッドは送信側のセットに結合される．

メソッド間の操作関係が，メッセージ送信・受信の両方である場合，結合規則(b)より，分離できない．

関係表現との対応：

＜分割条件5：メソッド間のメッセージ交換関係について＞
3.3 節のすべての操作関係を満たすことからすべてのセットは，partとなり得る．

このように分割や結合を行う際に用いた分割条件や結合規則の情報を用いて，関係表現によるオブジェクト構成のための指摘ができる．

---

†1　「partとなり得る」という表現は，そのセットが他との関係で部分オブジェクトになり得ない条件がなければ「partにする」ことを意味する．

4．支援機能の事例

本章では，これまでに述べた構成支援の機能について適用事例を挙げて説明する．本論文で示した支援方法は，２章での関係表現を備えたオブジェクト表現システム全般に適用できる．ここでは，支援事例を示すための表現システムとしてKORE/KR［9］を取り上げる．KORE/KR はPrologをベースに作成されたオブジェクト表現システムであり，supersetやsubpartsを明示的に記述してオブジェクトの関係を規定するものである．

KORE/KR が持っているオブジェクト間の基本的関係は，次の３つである．

(1)superset関係：クラスの包含関係を表す．

(2)subparts関係：オブジェクトの機能・構造的関係を表す．

(3)instance-of関係：表現されたクラスを利用するための実体（インスタンス）との関係を表す．

この他に，(a)meta-object関係，(b)manager機能を持っている．(a)は，(1)～(3)のようにユーザに開放されている関係ではなく，表現システム内部の組み込み機能(Meta-object)との関係である．したがって，ユーザがその機能や関係を自由に定義できない．ただし，このMeta-object が持つ機能を補うための機能付加［12］（Flavors などにあるafter daemons 的な方法）は可能である．(b)は，オブジェクトの動作に直接影響する関係ではなく，指定されたオブジェクトについて，その実行を記録・管理するための機能を実現する．KORE/KR は組み込みとしてManager クラスを持ち，このオブジェクトがmanager機能を実現している．

KORE/KR はこの他にインスタンスに関しての基本的な機能を実現するためのObjectクラスや，本稿の支援機能を実現するためのStructure-objectクラス等いくつかの組み込みがある．これら複数のオブジェクトを基本とし，開発・実行が進められる(Fig.2)．

Fig.2 挿入>

ObjectとManager はFig.２のようにsubparts関係で表現されている．したがって，ユーザ定義のクラスがObjectを継承することで，そのクラスのインスタンス作成と同時に，Managerのインスタンスが部分オブジェクトとして作成され，Manager クラスの機能を部分機能として利用できるようになる．組み込みのManager が持つ記録・管理の機能は，各オブジェクトについてのメッセージの送信・受信状況の情報や，スロットへのアクセス情報，スロットの状態変化の履歴などである．

KORE/KR の基本的な関係を挙げたが，この中で本支援機能の視点は，前述の(1)と(2)の関係である．他の関係は，表現レベルではなく，プログラムの実行や管理の際に必要となるものである．

4.1 節には，3.1 節のオブジェクト分割による構成支援の事例を示し，4.2 節で 3.2節に対応した実際的な規模のオブジェクトに対する構成支援例を示す．事例として示すオブジェクトは，すべて表現システムKORE/KR 上で実現したものである．

4.1 最小セットによる構成支援

本節で例に取り上げるクラスはFig.３の'house' である．このクラスの構成を図で表したものがFig.４(a)である．このクラスを，構成支援方法で分析する（オブジェクト表現を

F.3,4 挿入>

データとして渡して処理させる）と，Fig. 4(b)の構造にすべきことが指摘される．

構成支援方法では，対象クラスのシンタックスを分析して，構造的情報を取り出し，3章で示した規則や条件により，Fig. 5に示すような新しいクラス構成のための情報を提示する．このFig. 5は，'house' クラスが，二つの最小セットであるpart(a)とpart(b)をそれぞれpartオブジェクトとする，適切なオブジェクト構成に変更できることを示している(CLASS STRUCTURE INFORMATIONにおいて，各セットは'***** part'と示されている)．さらに，part(a)とpart(b)をsubparts関係のオブジェクトとして切り出した場合，house クラスに残るメソッドが示されている(INFORMATION ABOUT OTHER METHODS)．この場合，メソッドであるgoやenter は，メソッドとしてそれぞれ独立なものではなく，メソッドセットとしてまとまっていることがわかる．house クラスは，それ以外の，メソッドを持たないこともわかる．これらの指摘によって，Fig. 4(b)の構造が実現される．各オブジェクトの記述を，

F.5,6 挿入>

Fig. 6に示す．

4.2 オブジェクト分割による実際的な支援例

前節に示した事例では，最小セットのレベルで，partとして完全に分割できた．しかし，実際的な対象では，最小セットではpartとして分割できない場合がほとんどである［11］．

本節では，KORE/KR 上に作成された,Simplex法による多目的最適化のアルゴリズムを実現した'decision'クラスを事例として取り上げる．構成支援方法では対象クラスのシンタックスを分析し，構造的情報を取り出す．これらの構造的情報をもとに，3.1 節での最小セットとしての分割によるクラス構成の情報A(Fig.7) が導かれる．Fig.7-a が最小セット(SLOTSとMETHODSの対)であり，Fig.7-b がメソッドセットである．この情報Aでは，'decision'クラスが2つの最小セットといくつかのメソッドセットから成立っていることを示している(Fig.8)．しかし，この最小セットはすべてpartとして分割できない（Fig. 7では，すべての最小セットがmainと表示されている)．

F.7,8 挿入>

このように，構成支援方法によりセットをpartと判断できない場合は，各セットが3.3節の操作関係（特に(2)）を満足していないためである．支援情報(Fig.7-a）には，スロットについての規定2-2,2-3 やメソッドについての規定2-1 を満足していない情報が，それぞれKEY slots とKEY methods として提示されている．

KEY の意味は，「KEY slots:そのセットが内部処理を行うために，セット以外のスロットの参照を必要としている（分割条件３）．KEY methods:そのセットが内部処理を行うために，セット以外のメソッドを呼びだす必要がある（分割条件４）」である．

KEY を減らすために，基本セットとしての分割(3.2節）を行う．Fig. 9にその支援情報Bを示す．情報Bでは，3.2 節の規則(c)によりいくつかのメソッドセットが最小セットに結合され，基本セットを構成したため，KEY methods が無くなる(Fig.10)．この段階でも，KEY methods が残る場合，そのKEY となるmethodは，superset関係の記述の対象として考えられる．この例のオブジェクトでは，情報Bでもsubparts関係として分割できるセットが指摘されていない．partにするためのメソッドに関するKEY がないことから，各セット

F.9,10

は，メッセージを処理する機能としては閉じている．したがって，Fig. 7よりも，partに

挿入＞｜できる可能性が強い．

データ制御としてのKEY slots は，その制約が少ない場合はメッセージ交換時のargumentとして処理するようにメソッド等を変更することが検討できる．この変更により，partオブジェクトとしての構成を導ける．ただしこの場合の変更は，オブジェクトに対するメッセージに直接関わるため，ユーザの意思決定が必要となる．この変更を行った場合，Fig.11の関係図にある形で，基本セットSET Bをpartとして分離することができる．最終的には，構成支援方法によって，'decision'クラスをSET Aといくつかのメソッドセットで記述し，そのpartオブジェクトとしてSET Bが新たに構成できることがわかる．これは，Simplex 法のアルゴリズムの中で，併合挿入法という他のプログラムでも利用できる汎用的アルゴリズムの部分をpartとして取り出すことを示している．

F.11
挿入＞

5． 考察

構成支援の支援機能について前章の実例をもとに考察する．

4章での結果から，構成支援による効果は，以下のようにまとめられる．

○ 作成したオブジェクトの機能や属性を，まとまりとして把握できる．

○ 構成支援により得られる最小セットや基本セットを用いて，subpartsの関係表現を適切に導くことができる．

○ メソッドセットの情報により，superset関係による機能分散の検討やオブジェクト自身が持つ機能の検討ができる．

○ オブジェクトの関係表現の意味が表現システムで統一して実現される．

subparts関係で，オブジェクトとして新たに分割されたものや，各セットの情報は，後にオブジェクトが持つ機能の改良や再利用のために，重要な働きを担うものである．

4.2 節で，mainと指摘された基本セットに対して，KEY slots をメッセージ交換時のargumentとして処理することを取り上げたが，利用者にこれらの検討を進めさせ，関係表現の利用を促すことは，各オブジェクトの機能を明確にすることになり，再利用に対して効果的な記述を進めることになる．

この構成支援は，対象の機能や内部構造（スロットを中心にしたメソッドのまとまり方）を正確に把握できることから，オブジェクト開発と並行して利用することで，開発時のデバッグにも貢献できる．

たとえば，この支援過程におけるメソッドセットの情報を用いて，メソッドの必要性を検討できる．このメソッドセット情報から，あるメソッドがオブジェクト内部では全く利用されていない(SENDもRECEIVEもない)孤立したものであることが分かり，オブジェクト自身の機能を再検討することにより，不必要かどうかの判定ができる．Fig.7-b のINFORMATION ABOUT OTHER METHODS では，そのオブジェクトの機能として孤立したメソッドセット(make-weightとmake-weight1)が指摘されている．また，対象オブジェクト内での変更（特にメソッドの引数等の変更）の影響関係も把握できる．

実際には，対象に適した構造や記述は，本論文で提案する構成支援方法だけで規定できない．第4章の適用事例でも，表現の細かい変更までを自動に行わず，関係表現や記述の

検討すべき情報を利用者に提示することにとどめている．したがって，結果の評価は利用者の判断に任される．しかし，本論文で示した構造面だけでの情報でも，関係表現の指摘や関係表現の概念がシステムで統一され，改良・拡張・再利用が容易になるなど，支援機能として十分効果が得られる．

本論文では，分割の観点からsubparts関係を中心に関係表現の支援を試みた．しかし，オブジェクトの分割方法に加えて，オブジェクトの構成支援には，全体構成の中で機能的視点からの各オブジェクトの同一性あるいは，融合の検討も必要となる．これらに関しては，オブジェクトの分割概念から関係表現の利用が統一され，個々のオブジェクトの機能や他のオブジェクトとの関係構造が明確になれば，それらの関係情報を用いることで支援を行うことができる．そこで，今後は，共有性の立場から，superset関係を中心にした関係表現や，同等性などの支援を検討する．4.2 節で，基本セット内のKEY methods の情報によりsuperset関係の対象を限定したが，この関係の支援には，メソッドの働きに注目し，表面的な構造的情報だけでなく，さらに細かい分析が必要となる．

6． 結び

オブジェクト指向概念を導入した表現システムの，オブジェクト構成問題に対処するための構成支援方法について述べた．この基本は，オブジェクトが持つスロットやメソッド，及びそれらの利用状況を用いて，関係表現として最適なオブジェクト構成への支援を行う方法である．本方法により，関係表現を用いたオブジェクト表現が指摘される．このことから，既に，関係表現を利用して実現された対象を除き，対象が，オブジェクト指向プログラミングの利点を用いた記述を行えるかどうかの判定をする基準になる．したがって，オブジェクト表現としてのプログラミングスタイルを確立するための一方法と考えられる．

本論文の視点では，スロットに注目した支援がsubparts関係につながり，メソッドに注目した支援がsuperset関係につながることを示している．

最後に，本研究に関して貴重な議論，助言をして頂いた戸田部長，情報社会学室の新谷，平石研究員に感謝します．

尚，本研究は，第五世代コンピュータプロジェクトの一環として行ったものである．

[参考文献]

[1] Bobrow,D.G.,Kahn,K.,Kiczales,G.,Masinter,L.,Stefik,M.and Zdybel,F. : *COMMONLOOPS* Merging Common Lisp and Object-Oriented Programming,ISL-85-8, Xerox Palo Alto Research Center, 1985

[2] Bobrow,D.G.and Stefik,M. : *The LOOPS Manual*,Xerox PARC Knowledge-based VLSI Design Group memo KB-VLSI-81-13 1983

[3] Brachman,R.J. : What IS-A Is and Isn't:An Analysis of Taxonomic Links in Semantic Networks, *IEEE Computer* 16(10),(1983),pp.30-36

[4] Chikayama,T. : Unique Features of ESP, *Proc. International Conference on Fifth Generation Computer Systems* (1984) pp.292-298

[5] Furukawa,K.,Takeuchi,A.,Kunifuji,S.,Yasukawa,H.,Ohki,M.and Ueda,K. : Mandala:A Logic Based Knowledge Programming System, *Proc. International Conference on Fifth Generation Computer Systems*, (1984),pp.613-622

[6] Goldberg,A.and Robson,D. : *Smalltalk-80 The Language and its Implementation*, Reading Massachusetts, Addison-Wesley, 1983

[7] Harary,F. : *GRAPH THEORY*, ADDISON-WESLEY, 1972

[8] Hewitt,C. : Viewing Control Structures as Patterns of Passing Messages, *Journal of Artificial Intelligence* , Vol.8,No.3(1977),pp.323-364

[9] 片山：表現システムKORE/KR の概要，富士通㈱国際情報社会科学研究所, Brainware, 1985

[10] 片山：オブジェクト表現を用いたプログラム開発の支援環境－プログラム開発における計画・管理システムの実現－, *ICOT Technical Report* ,TR-169, 1986

[11] 片山：オブジェクト表現開発のためのクラス構成支援について，情報処理学会，知識工学と人工知能研究会，50-8 (1987) 87-AI-50

[12] 片山，新谷：知識テーブル（その利用)-知識ベースにおける対象指向表現とその処理機構の試作- ，第31回情報処理全国大会論文集,(1985),pp.1233-1234

[13] 片山，新谷，平石：問題解決支援環境KORE（その３）－知識表現サブシステムKORE/KR とその概要－，第32回情報処理全国大会論文集,(1986),pp.1147-1148

[14] Mizoguchi,F.,Ohwada,H.and Katayama,Y. : LOOKS:Knowledge Representation System for Designing Expert Systems in a Logic Programming Framework, *Proc. International Conference on Fifth Generation Computer Systems*,(1984),pp.606-612

[15] Moon,D.: Object-Oriented Programming with Flavors, *Proc. ACM OOPSLA Conference*, (1986),pp.1-8

[16] 垂水他：クラス再利用支援のためのオブジェクトモデル，コンピュータソフトウェ

ア, Vol.3 No.3(1986),pp.61-70

[17] Tokoro,M. and Ishikawa,Y.:A Concurrent Object-Oriented Knowledge Representation Language Orient84/K: Its Features and Implementation, *Proc. ACM OOPSLA Conference*, (1986),pp.232-241

[18] Weinreb,D.and Moon,D.: *Lisp Machine Manual*, 1981

[19] 横井, 福永：対象指向言語用知的プログラミング環境, 第31回情報処理全国大会論文集,(1985), pp.1229-1230

[20] Yonezawa,A.,Briot,J. and Shibayama,E.:Object-Oriented Concurrent Programming in ABCL/1, *Proc. ACM OOPSLA Conference*, (1986),pp.232-241

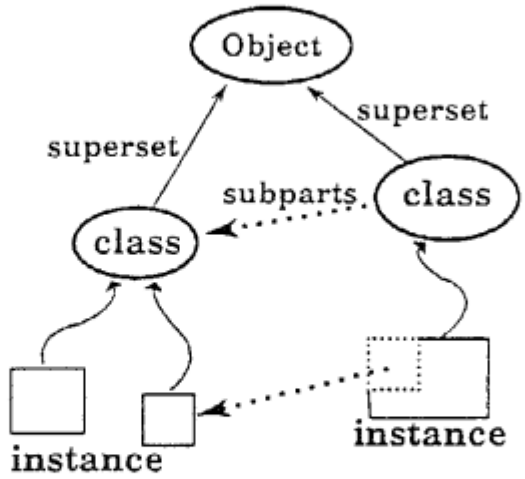


Fig.1 オブジェクトの関係表現

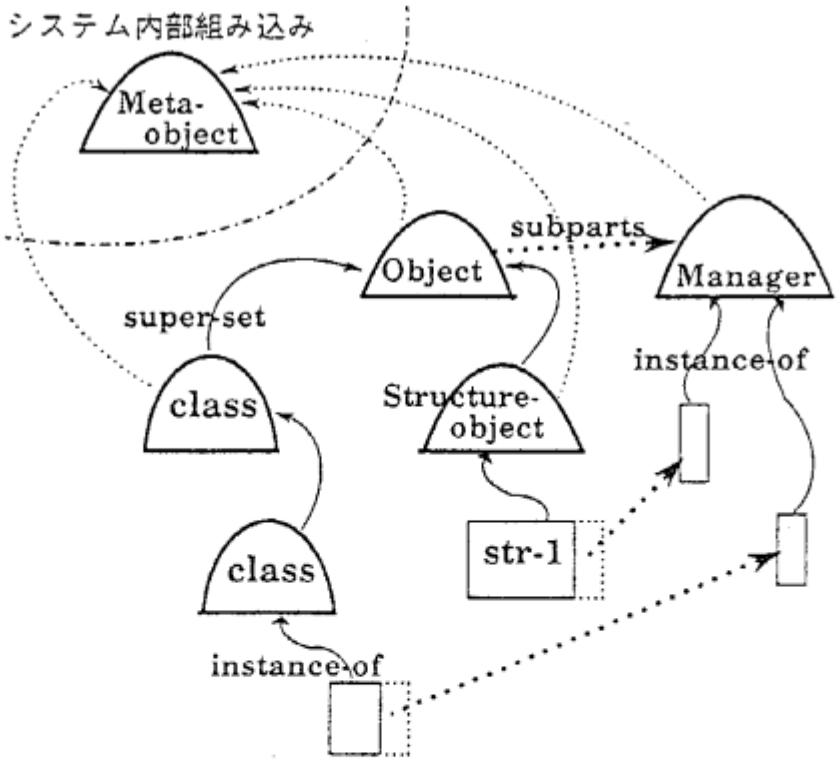
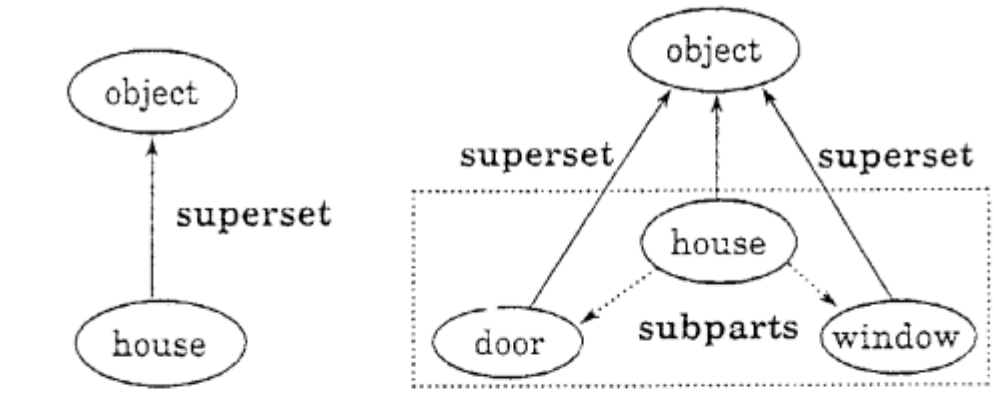


Fig.2 KORE/KRの基本オブジェクト構成

```
defclass house ::
 superset   object ;
 value_set  door - close,  window - shut ;
 method
    go    :=: ( @door=open, ... ; ...) ;
    enter  :=: ( @window=open, ...; ...) ;
  door_open    :=: ( @door=open, ... ;
                     #(door,open), ...) ;
  door_close   :=: ( @door=close, ... ;
                     #(door,close), ...) ;
  window_open :=:   ... ;
  window_shut :=: ( @window=shut, ... ;
                     #(window,shut), ...).
```

Fig.3 'house'クラスの記述



(a) 初期のクラス構成　　(b) 支援後のクラス構成

Fig.4 事例オブジェクトのクラス構成

```
*********  structure_object1   *********
THE STRUCTURE-DATA OF  house  CLASS ***
*************************************
****   CLASS STRUCTURE INFORMATION  ****
*************************************
          ----------------------------
          *****  part
             **SLOTS**
part(a)      door
             **METHODS**
           door_close/0 door_open/0
          ----------------------------
          *****  part
             **SLOTS**
part(b)      window
             **METHODS**
           window_shut/0 window_open/0
          ----------------------------
**********  end of data   **********
*************************************
*** INFORMATION ABOUT OTHER METHODS ***
*************************************
 ----------------------------
    **METHOD SET**
  go/0 enter/0
     GETVALUE
       window door
 ----------------------------
**********  end of data   **********
```

Fig.5 クラス構成の支援情報

```
defclass house ::
 superset   object ;
 subparts door1 - door, window1 - window ;
  method
    go :=: (door1 <- get,[state,State]),
           ( State=open, ... ; ... ) ;
    enter :=: (window1 <- get,[state,S]),
              ( S=open, ... ; ... ) ;
    door_open   :=: (door1 <- open);
    door_close  :=: (door1 <- close);
    window_open :=: (window1 <- open);
    window_shut :=: (window1 <- shut).

defclass door ::
 superset   object;
 value_set  state - close ;
  method
    open :=: ( @state=open, ... ;
               #(state,open), ...);
    close :=: ( @state=close, ...).

defclass window ::
 superset  object;
 value_set  state - close ;
  method
    open :=: ( @state=open, ...) ;
    shut :=: ( @state=shut, ... ;
               #(state,shut), ...).
```

Fig.6 支援後のクラス記述

```
*********  structure_object2   *********
THE STRUCTURE-DATA OF decision CLASS ***
****************************************
****   CLASS STRUCTURE INFORMATION  ****
****************************************
----------------------------
    *****  main
         KEY slots
           insertion
         **********
         KEY methods
           mapfunc/2 mapplus1/3
           d_step42/4 d_step52/4
           step2_min1/3 set/4
         **********
         RECEIVE
           init/0 start/0
SET A    **********

    **SLOTS**
    gamma walfa alfa wgamma value
    weight wh ws wo wl
    **METHODS**
  d_step51/4 step2_min/2 d_step4/0
  d_step41/4 mapplusd/5 aberage/3
  d_step6/4 ...
 ----------------------------
 ----------------------------
    *****  main
         KEY slots
           value
         **********
         KEY methods
           dev/3 mapsend/5 insertion1/4
         **********
         RECEIVE
           start/0
SET B    **********

    **SLOTS**
    table insertion
    **METHODS**
  insertion/2 insertion/0 merge/0
 ----------------------------
**********  end of data   **********
```

Fig.7-a クラス構成の支援情報A

```
****************************************
*** INFORMATION ABOUT OTHER METHODS ***
****************************************
 ----------------------------
    **METHOD SET**
  make_weight/3 make_weight1/5
 ----------------------------
 ----------------------------
    **METHOD SET**
  step2_min1/3
     RECEIVE
       step2_min/2
 ----------------------------
 ----------------------------
    **METHOD SET**
  insertion1/4
     RECEIVE
       insertion/2
 ----------------------------
 ----------------------------

         . . .

 ----------------------------
    **METHOD SET**
  init/0
     SEND
       step2/0
 ----------------------------
 ----------------------------
    **METHOD SET**
  start/0
     SEND
       merge/0 step3a/0 step3b/0
 ----------------------------
**********  end of data   **********
```

Fig.7-b クラス構成の支援情報A

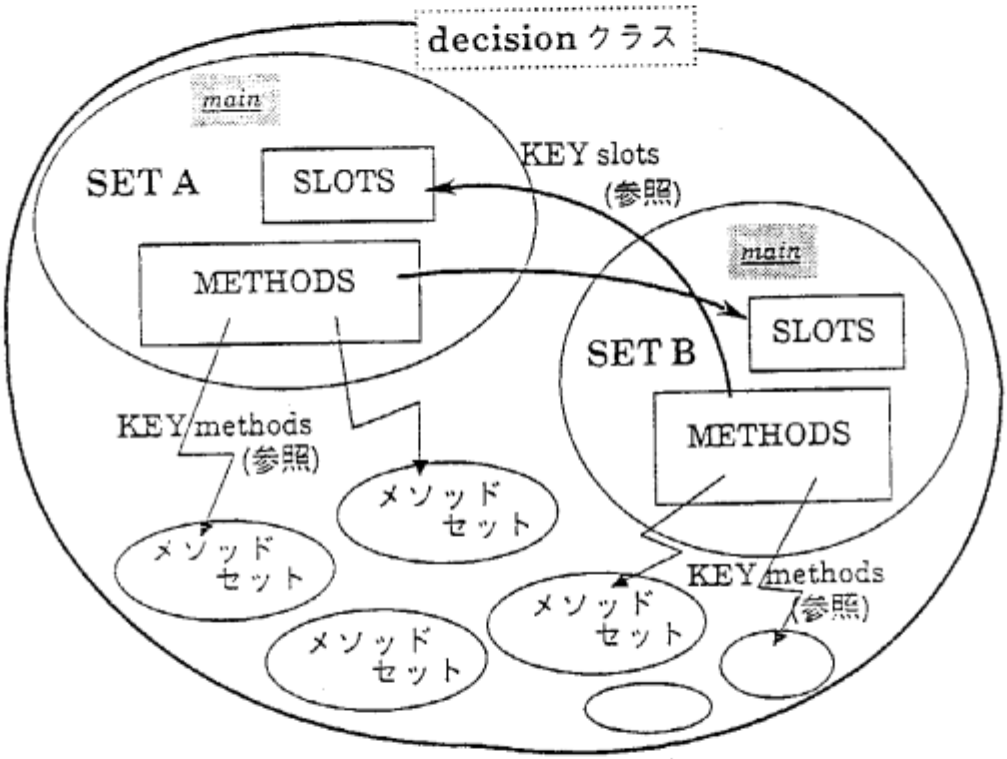


Fig.8 クラス 構成のための支援情報Aの関係図

```
*********  structure_object2   *********
THE STRUCTURE-DATA OF decision CLASS ***
****************************************
****   CLASS STRUCTURE INFORMATION  ****
****************************************
 ----------------------------
    *****  main
         KEY slots
           insertion
         **********
         RECEIVE
           init/0 start/0
         **********
SET A
    **SLOTS**
    gamma walfa alfa wgamma value
    weight wh ws wo wl
    **METHODS**
  set/4 d_step52/4 step2_min1/3
  mapfunc/2 d_step51/4 step2_min/2
  d_step4/0 d_step41/4 mapplusd/5
  aberage/3 d_step6/4 ...
 ----------------------------
 ----------------------------
    *****  main
         KEY slots
           value
         **********
         RECEIVE
           start/0
         **********
SET B
    **SLOTS**
    table insertion
    **METHODS**
  insertion1/4 mapsend/5 dev/3
  insertion/2 insertion/0 merge/0
 ----------------------------
**********  end of data   **********

****************************************
*** INFORMATION ABOUT OTHER METHODS ***
****************************************
 ----------------------------
    **METHOD SET**
  make_weight/3 make_weight1/5
 ----------------------------
 ----------------------------
    **METHOD SET**
  init/0
     SEND
       step2/0
 ----------------------------
 ----------------------------
    **METHOD SET**
  start/0
     SEND
       merge/0 step3a/0 step3b/0
 ----------------------------
**********  end of data   **********
```

Fig.9 クラス構成の支援情報B

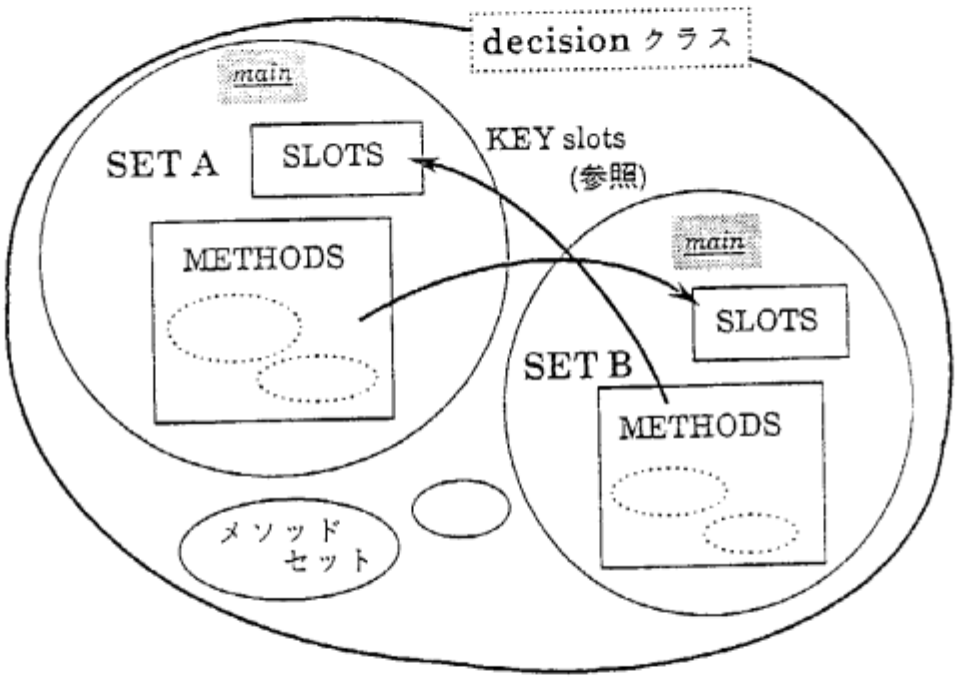


Fig.10 クラス構成のための支援情報Bの関係図

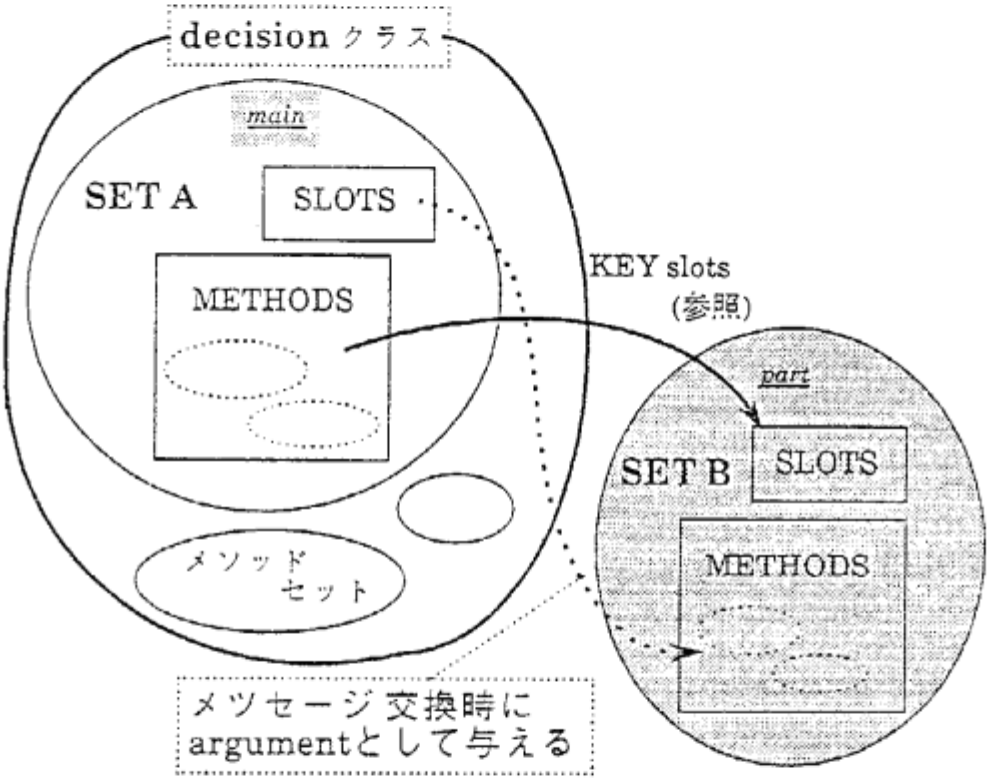


Fig.11 最終的な支援情報の関係図