データベースシステム 104−41
（1995. 7. 20）

ワークフロー管理システム WorkFlowBase における ワークフローデータモデル

国島丈生　　上林弥彦

奈良先端科学技術大学院大学 情報科学研究科　　京都大学工学部

オフィスにおける定型的協同作業を支援するグループウェアとして、あらかじめ定められた業務の流れ(ワークフロー)を電子メールなどを用いて計算機により支援するワークフロー管理システムが近年注目されている。このシステムでは業務の流れの定義や進捗状況、業務に関するさまざまな条件、業務の進行にともなって発生する資源(プロダクト)などをシステムで管理する必要があり、その構築には DBMS との連携が不可欠となる。我々は、業務の進行によって発生してくるさまざまな資源をも含めたデータ管理環境としての性格を持つワークフロー管理システム WorkFlowBase を提案している。本稿では、WorkFlowBase におけるワークフローモデルについて、制約の継承、ビュー機能など、データモデル的側面について述べる。

Workflow Model on a Workflow Management System WorkFlowBase

Takeo Kunishima　　Yahiko Kambayashi

NARA Institute of Science and Technology　　Kyoto University

Workflow Management Systems (WFMSs) are the groupware systems which support typical cooperative work in real offices. WFMSs must be closely connected with DBMSs since WFMSs manage several kind of data about supported works – their workflows, the states of progress, products of the works, etc. Now we are proposing a workflow management system WorkFlowBase, which manages workflow data with database functions. In this report, we explain workflow model on WorkFlowBase from the data model point of view. It has some features similar to those of data models: inheritance of constraints between workflow nodes, view functions, etc.

1 はじめに

オフィスにおける定型的協同作業を支援するグループウェアとして、業務の流れを計算機によって支援するワークフロー管理システム (Workflow Management System, WFMS) が近年注目されている [2,5] 。これは、あらかじめ定められた業務の流れ (ワークフロー) を、電子メールなどを用いて計算機により支援しようというシステムである。

ワークフロー管理システムでは、業務の流れの定義や進捗状況、業務に関するさまざまな条件、業務の進行にともなって発生する資源 (プロダクト) などをシステムで管理する必要があり、その構築には DBMS との連携が不可欠となる。しかし、現状のワークフロー管理システムは業務の流れの定義と各作業のルール化に重点がおかれており、DBMS との連携を充分に考えたものはほとんど見当たらない [3]。

我々は、業務の進行によって発生してくるさまざまな資源をも含めたデータ管理環境としての性格を持つワークフロー管理システム WorkFlowBase を提案している [6–8] 。これは、各活動を活動オブジェクトとして表現し、その集合としてワークフローを表現するものであり、構造や資源等の動的な変更、制約の継承機構などを持つ。また、一つのワークフローは対応する作業全体の手順・構造を表すため、ある条件に合致した作業のみを取り出してワークフローとして見たい、といった要求は自然なものであると考え、ワークフローに対するビュー機能を導入している。

本稿では、WorkFlowBase におけるワークフローモデルについて、制約の継承、ビュー機能など、データモデル的側面について述べる。以下、2章ではワークフロー管理システムの概要、DBMS との関連、現状のワークフロー管理システムの問題点などについて述べる。3章で、現在我々が提案している WorkFlowBase の基本概念と概要を述べ、4章で WorkFlowBase におけるワークフローモデルについて、5章でワークフローモデルにおける制約の継承について、6章でワークフローに対するビュー機能についてそれぞれ述べる。7章は本稿のまとめと今後の課題である。

2 ワークフロー管理システム

2.1 ワークフロー

Georgakopoulos ら [1] によると、ワークフローとは「あるビジネスプロセスを成し遂げるために組織化されたタスクの集まり (a collection of tasks organized to accomplish some business process)」である。個々のタスクは、人間やグループ、ソフトウェアなどによって実行される作業を表す。

通常、ワークフローにはタスクの実行に関する順序も定義されていることが多い。また、各タスクはさらに細分化されたタスクの集まりであってもよい。すなわち、ワークフローは一般に階層構造をなす。

論文の査読過程を表すワークフローの例を図 1に示す。

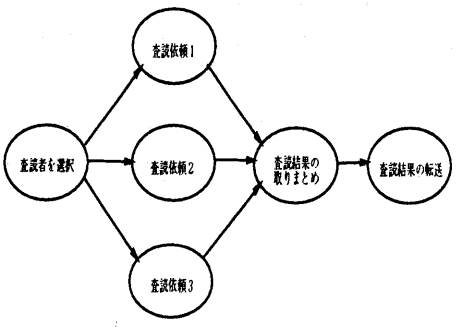

図 1. ワークフローの例 – 論文の査読過程

2.2 ワークフロー管理システム

ワークフロー管理システムは、ワークフローを用いて定義された業務を計算機により支援しようというシステムである。支援の方法は以下のように大別される [2]:

- ツール間のデータの受渡しを自動化する
- 電子メールを用いて利用者間のデータの流れを支援する (電子回覧)
- DBMS を用いて業務の進行過程で発生したデータを管理する

ワークフローにおける各タスクを実行する作業主体 (以降、これをエージェントと呼ぶ) は一般に異なってもよく、その意味から、ワークフロー管理システムはグループウェアの一例として分類される。また、ワークフロー管理システムは分散環境において動作することが前提とされる。

2.3 従来のワークフロー管理システムの問題点

2.3.1 システムの柔軟性

グループウェアの視点から指摘されているのは、柔軟性に欠けるという点である。

すなわち、一旦業務がワークフローの形で定義され運用されてしまうと、フローを変更することが難しくなるという点である。例えば、一旦完了してしまった業務に対して差戻しやフローの変更を行うと、現在進行中の業務にまで影響が及ぶ。この影響範囲を適切に定めることが難しい。あるいは、業務を行う人間が出張などの理由で業務ができない場合、先に業務を進めておきたい場合もある。これは例外処理に相当する。また、必要な資源・分担者など、活動の詳細が事前には決まっていなくてもよい場合もある。

WFMS が対象としているのは主に定型業務であるのでこのような変更は比較的少ないと考えられているが、現実には、頻度は少ないが定型業務でも変更が起こる。そのため、商用の WFMS でも、記述したワークフローの柔軟性は問題になっており、大抵の WFMS ではフローに対する例外処理を記述できるようになっている。しかし、業務の差戻しなどに対して充分な機能を提供している WFMS はまだない。

2.3.2 データ管理環境との統合・連携

WFMS では、作業そのものの手順を計算機によって管理する。そのため、データベースなどのデータ管理環境との統合・連携が不可欠である。

WFMS で管理すべきデータとしては、作業に関するデータ (プロセスに関するデータ)、作業に必要なデータや作業の結果生成されるデータ (プロダクトに関するデータ) の 2 種類がある。これらを統合して管理し、検索等のアクセス手段を用意することが必要になる。現在の商用 WFMS では、プロセスデータは管理しているが、プロダクトデータをも統合して管理している例は少ない。

3 ワークフロー管理システム WorkFlowBase

WorkFlowBase では、ワークフローを、ひとまとまりの作業を表すオブジェクトの集合とそれらの間の制約集合とからなるものとして扱い、データベース技術を用いてワークフロー管理に必要な機能を実現する。

WorkFlowBase の目標は、データベース技術の応用により 2.3節に述べたワークフロー管理システムの課題、すなわち、ワークフローに柔軟性を持たせること、作業に関係する資源も含めた統合的なデータ管理環境の提供、などを実現することである。

また、一旦定義したワークフローをデータベースに蓄積することにより、過去に定義したワークフローの再利用も可能になると考えている。

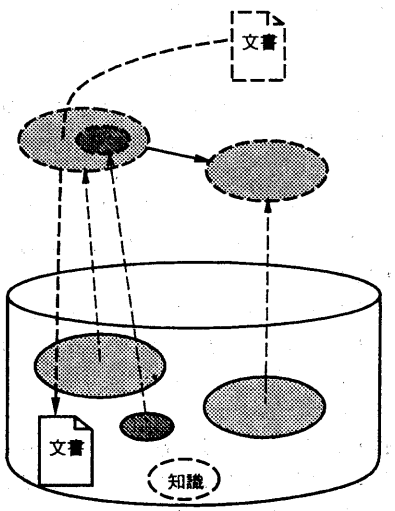

図 2. WorkFlowBase のシステム概要

4 WorkFlowBase におけるワークフローモデル

ここでは、[6-8] で提案したワークフローモデルの概略について述べる。このモデルでは、図 3 に示すように、ひとつの単位となる活動を活動オブジェクトとして表現し、活動オブジェクトの集約階層として一つの作業フローを表現する。

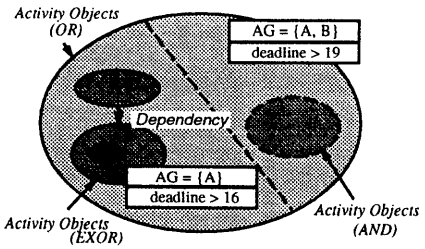

図 3. 活動オブジェクトによる作業フローの表現

活動オブジェクト o は、次の7つ組で表現される:

$$o = \langle I, O, AO, E, ATTR, AG, C \rangle$$

ここに、I はその活動に必要な入力オブジェクトの集合、O は活動の結果生じる出力オブジェクトの集合、AO は o の副活動を表す活動オブジェクトの集合、$E \in \{AND, OR, EXOR\}$ は副活動オブジェクト間の実行モード、$ATTR$ はその活動の属性集合、AG はその活動オブジェクトを実行するエージェントの集合、C は $I, O, AO, ATTR, AG$ に関する制約集合をそれぞれ表す。o と $o' \in AO$ には親子関係があるという。$I, O, AO, E, ATTR, AG, C$ などは動的に変更され得る。

活動オブジェクト o, o' について、$x \in O$(O は o の出力オブジェクトの集合) かつ $x \in I$(I は o' の入力オブジェクトの集合) が成り立つ場合、o' は o に依存するという。

活動オブジェクトの実行順序は、それらに依存関係がない限り特に規定しない。また、活動オブジェクトで表現される活動をいかにして実行するかはエージェントが決定し、活動オブジェクト中には記述しない。最終的に O の要素がすべて得られ、かつそれらが C で表される制約を満たしていれば、活動は完了したものとする。

5 活動オブジェクトの集約階層上での属性・制約の継承

5.1 継承機構

集約階層によって活動を表現する場合、親オブジェクトの属性やその制約が子オブジェクトでも有効になることがある。また、エージェント集合については、親オブジェクトでの値が子オブジェクトでも有効となることがある。例えば、下位の活動を実行するエージェントが決定されていない場合、上位の活動を実行するエージェントが一時的に代行するようなことがある。

以上のような、親オブジェクトから子オブジェクトへの属性・制約の伝搬を表現するため、集約階層上で親子関係にあるオブジェクト間での継承機構を導入する。

属性については、親オブジェクトに定義されている属性のうち、ガードされない属性のみ継承される。ガードについての詳細は5.3節に述べる。また、エージェント集合については、子オブジェクトで未定義であれば無条件に親オブジェクトから継承する。

制約については、以下のように継承が行われる。一般には、ある活動オブジェクト o の制約 $c \in C$ は、o の属性と o の子オブジェクトの属性とを変数として持つ述語になるが、C のうち、o の属性のみに関する制約 (自己制約) のみ子オブジェクトに継承される。

なお、活動オブジェクトの構造は動的に変更され得るため、継承は動的に決定される。

5.2 属性の制約の再定義

制約に関しては、親オブジェクトで定義された制約が変更されて子オブジェクトにおいて有効となることがある。例えば、下位の活動のデッドラインは上位の活動のデッドラインよりも前に設定される、というような場合がある。

これを解決するため、親オブジェクトから継承された制約を子オブジェクトで再定義できるようにする。

定義 1 活動オブジェクト o, o'(ただし o' は o の子オブジェクト) とその制約集合 C, C' において、$c \in C$ が C' に継承され、かつ c, c' が次の条件を満たすとき、c は c' で再定義可能という。

1. $c' \models c$
2. $\{c'\}$ と $C' - \{c\}$ が矛盾しない

□

例えば、活動オブジェクト o で制約 '$deadline \geq$ 19日' が定義されており、かつ他に属性 $deadline$ に関する制約がないものとする。このとき、制約 '$deadline \geq$ 16日' では再定義可能であるが、制約 '$deadline \geq$ 20日' では再定義可能でない。

親オブジェクトから継承された制約 c が制約 c' で再定義可能な場合は、無条件に再定義できる。一方、再定義可能でない場合は、以下の2つの解決策のうちいずれかを選ぶ。

- 親オブジェクトを実行するエージェントとの交渉によって再定義可能にしてから再定義を行う。
- 期限を設け、それまでに再定義可能でない状態を解消するという条件付で再定義を行う。この場合、親オブジェクトの制約と子オブジェクトの制約の間に矛盾が残るが、その矛盾は子オブジェクト内の制約の変更により期限内に解消しなければならない。親オブジェクトを実行するエージェントとの交渉は行えない。

5.3 継承のガード

ある活動オブジェクトに特有の制約は、子オブジェクトに継承しないほうがよい場合がある。また、親オブジェクトからの継承を、再定義するのではなく止めたい場合もある。

このような継承のガードをしたい状況は、以下の2通りに大別できる。

1. 上向きのガード (親からの継承を禁止)
2. 下向きのガード (子供への継承を禁止)

このうち1は、ガードする制約を恒真述語 $true$ に再定義することに相当する。そこで、前節で述べた方法にしたがって無制限にガードを付けることを防ぐ。2のガードをつけることには制限を設けない。また、属性のガードは2でしか行えない。

5.4 活動オブジェクトの活性化と継承との関係

現実の非定型業務では、ある活動を行うことを予測して、実際にその活動が始まる前にその一部を実行し始める、ということが行われる。

しかし、4章で述べた活動オブジェクトの集約階層は活動の構造的側面を表現したものであり、実行順序については規定していない。また、実行され始めていない活動オブジェクトからの属性・制約の継承が起こる。そこで、WorkFlowBaseのワークフローモデルでは、活動オブジェクトの活性化の概念を導入している。

活動オブジェクトに対応する活動が始められると、活動オブジェクトは活性化される。また、対応する活動が終了すると、活動オブジェクトは非活性化される。

入力オブジェクトが揃っていることと活性化状態とは同値とはしない。つまり、入力オブジェクトが揃っていなくても、活性化し、活動の一部を始めても良い。また、非活性状態にある活動オブジェクトの子オブジェクトだけを活性化して、部分的に活動を始めることも可能にする。

属性・制約の継承は活性化された活動オブジェクトのみを対象とする。すなわち、非活性状態にある活動オブジェクトには継承は起こさない。また、非活性状態にある活動オブジェクトから属性・制約が継承されることもない。

6 活動オブジェクト階層のビュー

ある活動オブジェクト階層に対して、条件を満たす活動オブジェクトのみを選択し、包含関係、実行順序、制約等を保存しつつそれらを階層的に構成した活動オブジェクト階層を、元の活動オブジェクト階層のビューと呼ぶ。本章では、このビューの構成方法について述べる。

6.1 推移的包含関連・依存関連

ビューを構成する際、活動オブジェクト間の包含関連、依存関連を保存するめには推移的な包含関連、依存関連を考慮する必要がある。例えば、活動オブジェクト $o_1, o_2, o_3 (o_2 \in o_1.SO, o_3 \in o_2.SO)$ において o_1, o_3 がビューに含まれる活動オブジェクトとして選択された場合を考えると、ビュー上では o_3 が o_1 の子孫活動オブジェクトになっていなければならない。また、$o_1 \rightarrow o_2, o_2 \rightarrow o_3$ であってビューに o_1, o_3 が選択された場合には、o_3 が o_1 に依存しているという関連が成立していなければならない。さらにこの場合、o_1 と o_3 との間に別の作業が入り、時間的に連続して作業が行われることはない、ということもビュー上で表現できる方が望ましい。

そこで、活動オブジェクト間の関連として新たに推移的包含関連、推移的依存関連を導入する。

定義 2 活動オブジェクト o, o' 間の**推移的包含関連** ($o \in^* o'$) を以下のように定義する:

1. $o \in o'$ ならば $o \in^* o'$
2. $\exists o''\ s.t.\ o \in^* o'',\ o'' \in o'$ ならば $o \in^* o'$
3. これ以外に $o \in^* o'$ が成立することはない。

また、o, o' 間の**推移的依存関連** ($o \overset{\pm}{\rightarrow} o'$) を以下のように定義する:

1. $\exists o''\ s.t.\ o \rightarrow o'',\ o'' \rightarrow o'$ ならば $o \overset{\pm}{\rightarrow} o'$
2. $\exists o''\ s.t.\ o \overset{\pm}{\rightarrow} o'',\ o'' \rightarrow o'$ ならば $o \overset{\pm}{\rightarrow} o'$
3. これ以外に $o \overset{\pm}{\rightarrow} o'$ が成立することはない。

□

ビューに選択された活動オブジェクト o, o' が元の活動オブジェクト階層上で $o \in^* o'$ が成立していたなら、ビュー上でも $o \in^* o'$ が成立するようにビューを構成する。$\rightarrow, \overset{\pm}{\rightarrow}$ についても、同様にビュー上でもこれらの関連が保存されるようにビューを構成する。

6.2 実行モードの保存

ある活動オブジェクト o について、$o.SO$ の実行順序は o の実行モードによって規定される。したがって、$o.SO$ の要素がビューに含まれるが o がビューに含まれないような場合でも、実行順序の意味が保存されるようにビューを構成する必要がある。

活動オブジェクト間の実行順序の意味をビュー上でも厳密に保存しようとすれば、それらの祖先の活動オブジェクトをすべてビューに選択しなければならない。しかし、これでは、最悪の場合元の活動オブジェクト階層とビューとが完全に一致してしまう。

活動オブジェクト o, o' 間のビュー上での実行順序は、元の活動オブジェクト階層上でのこれらの least upper bound (以下 lub) となる活動オブジェクトの実行モードによって決定されると考えると、o, o' の lub のみをビューに含めれば良く、また lub の実行モードが AND の場合は意味が完全に保存される。

ただし、lub の SO 中にビューに選択された活動オブジェクトを持たないものがあり、かつ lub の実行モードが OR や $EXOR$ である場合、これだけでは保存されない可能性がある。例えば、$o + o' + o''$ で o, o' が選択された場合、o, o' がともに実行されないようなケースが抜け落ちる。また、$o \oplus o' \oplus o''$ で o, o' が選択された場合、o'' の実行中は o, o' のどちらも実行されないが、この意味が保存されない。以上のような問題は、ビュー上での lub の SO に新たに活動オブジェクトを一つ追加することにより回避できる。

6.3 継承の展開

上位の活動オブジェクトがビューに含まれない場合に、元の活動オブジェクト階層上でその活動オブジェクトから継承されている属性や制約がビュー上では継承されなくなる。

この問題は、ビューを構成する際に、元の活動オブジェクト階層上で継承されてくる属性や制約をすべて展開し継承関係を解消することにより解決できる。

6.4 ビュー構成アルゴリズム

以上のような考察に基づくと、ビューの構成方法は次のようになる:

1. 包含関連に基づく活動オブジェクト階層の構成

   元の活動オブジェクト階層から、与えられた条件を満たす活動オブジェクトを選択する。それらを元の活動オブジェクト階層上での推移的包含関連を保存するように、活動オブジェクト階層として構成する。

2. 実行モードの処理 (6.2節)

   選択された任意の 2 つの活動オブジェクトについて、元の活動オブジェクト階層上でそれらの lub となる活動オブジェクトもビューに含める。lub の実行モードが $OR, EXOR$ であり、かつ lub の子に選択された 2 つの活動オブジェクトを子孫に持たないものが存在するとき、ビューを構成する活動オブジェクトとして、新たに lub の子を一つ付け加える。

3. 依存関連・推移的依存関連の保存

元の活動オブジェクト階層上で $o \rightarrow o'$ があり、o, o' がともにビューに含まれる場合、ビュー上でも $o \rightarrow o'$ をつける。また、元の活動オブジェクト階層上で依存関連 $o \rightarrow o_1 \rightarrow o_2 \rightarrow \cdots \rightarrow o_n \rightarrow o'$ があり、ビューに o, o' のみ含まれるような場合、新たに o'' という活動オブジェクトをビューに含め、$o \rightarrow o'' \rightarrow o'$ とする。

4. 継承される属性・制約の展開 (6.3節)

   1で選択された活動オブジェクトについて、もとの活動オブジェクト階層上でその上位の活動オブジェクトに定義されていて継承されてくる属性・制約をコピーし、継承関係を解消する。

図4に、上記のビュー構成アルゴリズムに基づくビューの構成例を示す。

7 おわりに

属性や制約は、その活動を実行するエージェントや、実行環境などからも影響を受けると考えられる。あるいは、ここでは活動はトップダウンに構成されることを仮定しているが、現実にはボトムアップに活動を構成することもある。これらを踏まえた形式化をさらに検討していきたいと考えている。

データモデル的視点からは、次のような課題があると考える:

- 制約の一貫性

  ワークフロー中に定義されたさまざまな制約の一貫性をいかにして保持するかはワークフロー管理システムの重要な課題の一つである。とくに、ワークフロー管理では、over constraints の状態から制約を選択して充足させる場合に制約の一貫性が問題になる。制約充足の方法、あるいは上位のワークフローと下位のワークフロー間での制約の干渉などについて検討する必要がある。

- 個人空間と共有空間

  協同作業では、共有情報の他に各個人が情報を蓄える機能も必要であり、かつ個人の作業環境と協同作業環境は融合されている必要がある [4]。ワークフロー管理に基づく協同作業に適した個人 / 協同作業空間のモデルについて、データモデル的な特徴を含めて明らかにしていく必要がある。

参考文献

[1] D. Georgakopoulos, M. Hornick, and A. Sheth. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, Vol. 3, No. 2, pp. 119–153, Apr. 1995.

[2] 松井. 業務の連携を自動化 時間短縮と管理を実現 - ワークフロー管理ソフトが日本でも利用可能に. 日経コンピュータ, No. 1994/5/2, pp. 57–67. 日本経済新聞社, 1994.

[3] 吉府, 田渕, 垂水. ワークフローとデータベースの相互連携システム. 研究会報告 GW9-23, 情報処理学会, Jan. 1995.

[4] 市村, 松下. 発言と行動の管理に基づいた協同作業支援電子メイル PilotMail. 情報処理学会論文誌, Vol. 33, No. 7, pp. 955–963, July 1992.

[5] 川上. ワーク・フロー管理が仕事を変える、組織を変える. 日経コンピュータ, No. 1992/9/21, pp. 56–75. 日本経済新聞社, 1992.

[6] 国島, 上林. 集約階層をなす協調作業上における作業間の属性の継承. 情報処理学会第 47 回全国大会予稿集, 第 6 巻, pp. 283–284, Oct. 1993.

[7] 国島, 上林. ワークフローモデルにおけるビュー機能. 情報処理学会第 49 回全国大会予稿集, 第 6 巻, pp. 247–248, Oct. 1994.

[8] 国島, 上林. 活動オブジェクトに基づく作業フローモデルの実行手続き. 情報処理学会第 48 回全国大会予稿集, 第 6 巻, pp. 241–242, Mar. 1994.

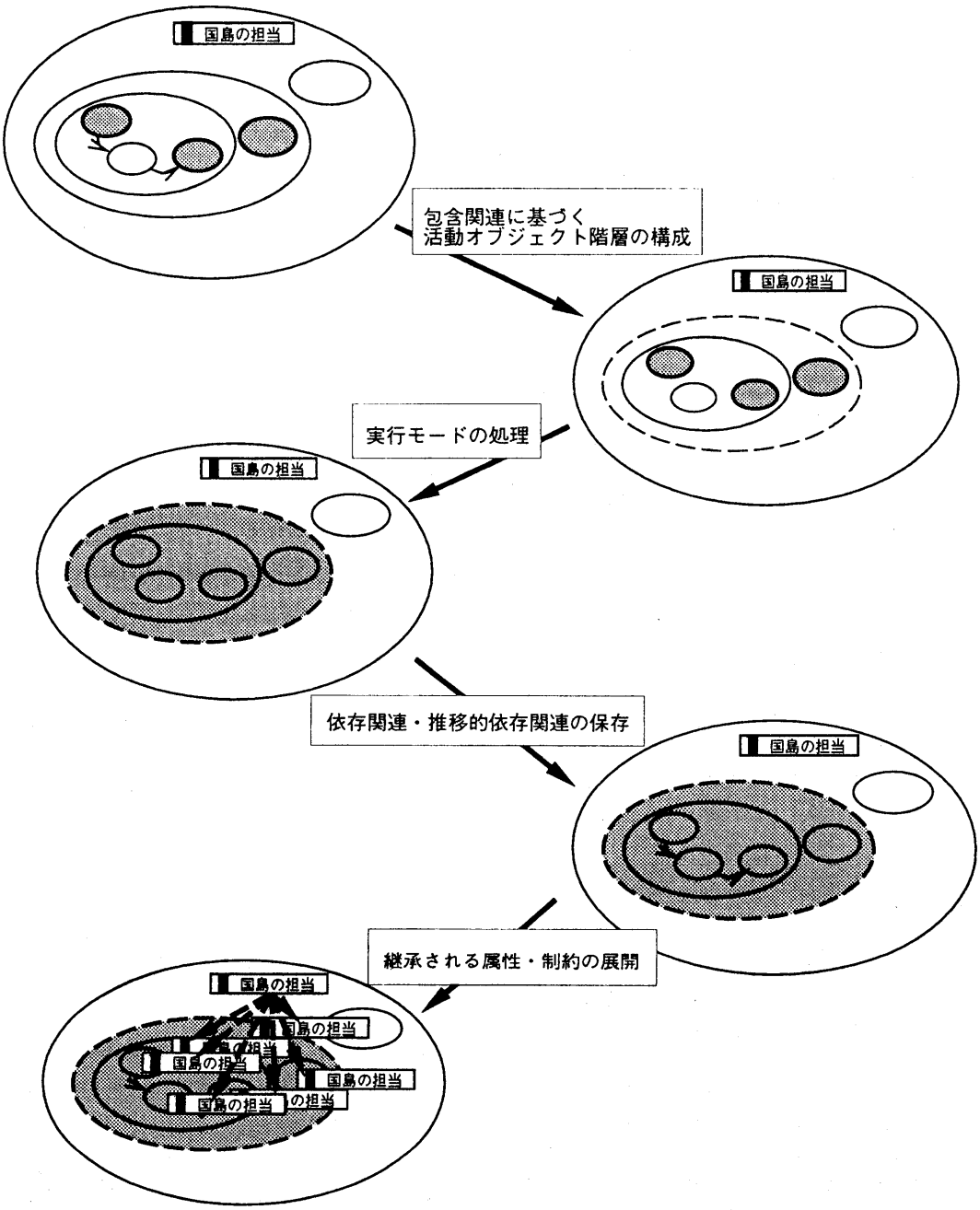

図 4. WorkFlowBase におけるビュー構成の例