ソフトウェア工学 99−15
（1994．7．8）

Realな動作の実現に必要なRoleの役割

片山　佳則
（株）富士通研究所　情報社会科学研究所

オブジェクト指向プログラミングに対する新しいモデルとして提案しているS-R-Oモデルにおいて，ロールの働きを改良／拡張することで，これまで困難であったトランザクションの管理がオブジェクトの振る舞いと矛盾なく扱えることを示す．具体的には，ロールのダイナミックな割り振りの実現とロールの記述内容内容に振る舞い起動のトリガーや制約条件を与えることで，オブジェクトの状態や振る舞いを管理する手段を実現するものである．

本稿では，これらの手段を用いて，S-R-Oモデルによって現実世界のプログラミングに適応可能なトランザクションメカニズムがどのように実現できるかを述べる．

Role Description for Realistic behaviour
- Scenario-Role-Object Model -

Yoshinori Katayama

FUJITSU LABORATORIES LTD.
Institute for Social Information Science (ISIS)
9-3,Nakase 1-chome, Mihama-ku, Chiba-shi, Chiba, 261 Japan

In the last years, the Object-Oriented methodology has been adopted at the level of programming and design. However, this methodology does not lead a perfectly reasonable mechanism. We have already proposed Scenario-Role-Object Model that worked out a number of problem on Object-Oriented Programming.

This paper defines the concepts of transaction on Object-Oriented Programming, and shows that Scenario-Role-Object Model has grouped actions into transaction. The Model is focused on the behavior of the object set. The atomic action is being substituted for Object's behavior by Role unit. We apply Scenario-Role-Object Model to modeling of the realistic behavior.

1．はじめに

これまでに我々は，オブジェクト指向プログラミングの分野において，オブジェクトの持つ振る舞いに関して信頼性／確実性などの保証を与えるために必要な技術を幾つか提案してきた．具体的には，オブジェクトの決定支援技術[1]から，オブジェクトの再構成技術[2]や部品化技術[3]などの提案である．個々の技術は，作成されたオブジェクトの再利用を進める技術の確立を一つの共通の目標としてまとめることができる．現在は，これらの技術を基盤として，オブジェクト指向プログラミングの利点を引き出させるための新たなモデリング技術としてS-R-Oモデルを提案し，その構築技術の定式化を進めている[4]．我々が提案しているS-R-Oモデルは，オブジェクト指向プログラミングの特徴を引き継ぎ，幾つかの問題点を捕えた上で，より現実的なプログラミングを進めるために必要な概念を導入し，拡張を行なったものである．

本稿では，S-R-Oモデルの一側面であるRoleユニットの働きを中心に，現実世界の問題をプログラムにマッピングさせるプログラミング技術がどのように変化し，何が実現可能になるかについてまとめる．

2．S-R-Oモデルの概要

2.1　S-R-Oモデルの基本概念

我々はオブジェクト指向プログラミングにおける幾つかの問題を解決した新しいモデルとしてS-R-Oモデルを提案している[4]．これは，オブジェクト指向の考え方にシナリオとロールという概念を導入して拡張したモデルである．S-R-Oモデルは，オブジェクトの再利用／保守を考慮したうえで，特に複数のオブジェクトの動作の関連を把握させることに焦点を当て，オブジェクトに対する外部環境を，オブジェクトに直交した形でロールとして記述することを基本とする．

その基本的立場は，対象領域側の記述をシナリオとロールが受け持ち，共有領域側をオブジェクトが受け持つことで，対象に依存するプログラムと依存しないプログラムを積極的に分けることにある．ロールには対象領域のために必要な対象依存プログラムや制約条件を記述する．シナリオでは，共有領域側で与えられるオブジェクトに対象領域側であるロールを割り振る役目を行なう．各オブジェクトは，割り振られるロールによって実行できるメソッドが決められる．ロールやシナリオが各対象領域ごとの特殊なプログラムを受け持ち，オブジェクトが

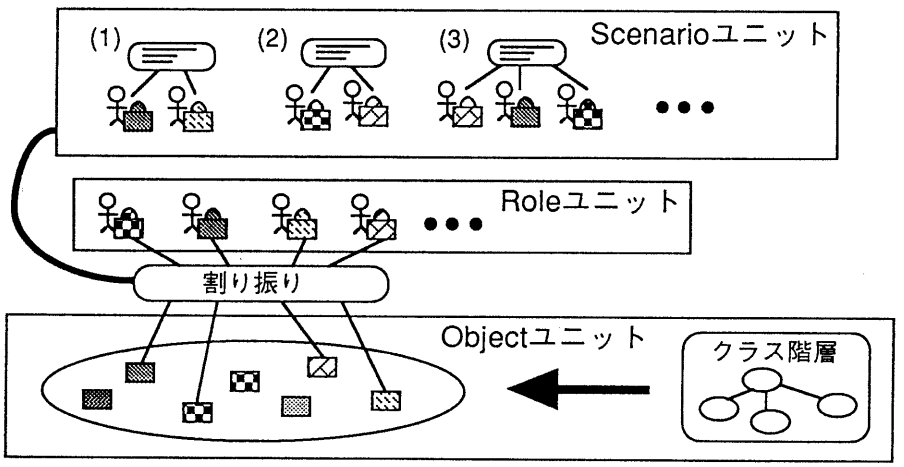

図１S-R-Oモデルの概念図

持つプログラムは，その対象領域に依存することなく広く利用できるものとする．このS-R-Oモデルの概念図を図1に示す．図のように，Objectユニットで与えられたオブジェクトにRoleユニットのロールを割り振る．シナリオではロールが割り振られたオブジェクトを基本モジュールとして用いる．このシナリオで記述するオブジェクトへのロールの割り振り関係を拡張することで，ダイナミックなロールの割り振りを実現でき，これにより，オブジェクトの振る舞いがロールの管理の元で柔軟な働きを実現できる[5].

2.2 S-R-Oモデルの構成要素

Objectユニット：S-R-Oモデルでは，各オブジェクトには，対象システムに依存しない，本来そのオブジェクトが持つべき情報（共通利用可能な振る舞いと状態変数）だけを記述する．つまり，基本的にオブジェクトには外部環境に関するデータを内在させない．個々のオブジェクトは，(1) メッセージを受け取ることで動作を実行する．(2) 性質の継承も，オブジェクトの関係構造に従う．これらの規約は，これまでのオブジェクト指向プログラミングのオブジェクトと変わらない．S-R-Oモデルに基づくプログラミングでは，その対象システムで規定される各オブジェクトの状況の変化や動作のトリガー，制約情報などの記述はすべてオブジェクトとは独立にロールを用いて記述する．これによって，従来はオブジェクトに分散していた対象システムに関わる記述はすべてオブジェクトから排除する．従って，S-R-Oモデルのオブジェクトのほとんどは，ボトムアップ的に用意することができる．さらにこのオブジェクトに割り振るロールやロールの推移関係を参照することで，対象システムが必要としている個々のオブジェクトどうしの振る舞いの関連や連係，状況に応じて起動される振る舞いなどを，メッセージで起動することなく把握できる．また，対象システムとしての外部環境をロールとしてオブジェクトから独立化させたことで，オブジェクトに対する制約をダイナミックに変更することも可能になる．

Roleユニット：ロールでは，オブジェクトに対する様々な外部環境の規定を行なう．その中でも特に，現在ではオブジェクトの横のつながりを振る舞いの関係として捕えることが基本となっている．

従来のオブジェクト指向プログラミングでは複数のオブジェクトの動作関係を理解するためには，実際にメッセージを送信し，起動して，その動作内容を詳細に辿らなければならなかった．しかしながら，S-R-Oモデルでは，従来のオブジェクトの記述を極力コンパクトにし，振る舞いの起動のトリガーや各振る舞いの実行に対する制約の割り振りをオブジェクトと異なる軸であるロールによって独立に管理する．このことにより，複数のオブジェクト間の動作の関係は，ロールの記述を見ることによってメッセージでの起動以前に理解でき，推論できるようになる．さらにこのロールの推移構造は，オブジェクトが行なう振る舞いに対する操作の合理性を判断する基本材料になり，複数のオブジェクトの動作の切れ目を与える指標になる．

Scenarioユニット：ロール自身は，対象システムの中で扱われる各オブジェクトの立場を明確にするものであり，実体ではない．このロールは，オブジェクトと結び付けられることではじめてその意義が生まれる．対象システムから見れば，オブジェクトだけでもロールだけでも不完全であり，これらが結合することで始めて対象システムの有効なモジュールを実現する．シナリオは，共有部品としてのボトムアップviewであるオブジェクトと対象システムとしてのトップダウンViewであるロールを対象システムの立場から対応付けるものである．つまり，シナリオは対象システムのために必要なモジュールを構成するためとそれらのモジュール

を用いて実際の対象システムの動作を実現させることが主な働きになる．オブジェクトやロールは，個々に再利用できることを前提とするために，それぞれの記述には独立性を保たせている．特にロール側は，より抽象度の高いキーワードで起動のトリガーや制約の記述を行なうこととし，広く利用可能にさせている．それらのキーワードを対象システムとして実際にオブジェクト側のどの部分に対応させるかを決定することがロールの割り振りであり，シナリオの重要な役目である．シナリオでのこの対応のわずかの変化によって，対象システムとしての機能に大きな違いが生まれる．S-R-Oモデルではシナリオがオブジェクトとロールの再利用を促進する機構を実現する．

3．Roleユニットの役割

S-R-Oモデルに限らず，オブジェクト指向プログラミングでも，その表現の汎用性やモデル化技術によって，現実世界の問題や状況自身をモデルの対象として扱うことができる．しかし，現実世界の操作を実現するためには，モデルの中での各操作の合理性を保証する必要がある．個々のオブジェクトはその記述段階ではそれぞれ独立に正当性が保たれている．しかし，動作の進行によりオブジェクト間に生まれる制約関係を把握しにくくなり，操作の合理性を管理できなくなる．これを回避するために，各操作に対して確実な保証を与えられるようなトランザクションの概念の定義が必要になる．データベースの分野でも，ある利用者がアクセスするデータの正しさをはっきりさせるためにトランザクションの概念が定義されている．この概念により，物理的だけでなく論理的な意味でのロック等のメカニズムが適用され，様々な手続きの正しい実行を可能にしている[6].

S-R-Oモデルでは，これらの現実世界の操作への対応におけるオブジェクト指向プログラミングの立場での解決として，次の２つの方法をポイントとしている．

(i) Roleユニットが，オブジェクトと独立にオブジェクトがどのように動くべきかの指針を与え，管理するメカニズムを与える．

(ii) Scenarioユニットがオブジェクトとロールの割り振りを実行時にダイナミックに行なえるようにし，振る舞いの柔軟性を高める．

この両方を実現することで，モジュールは，対象システムが置かれている状況の変化に柔軟に対応し，必要に応じたダイナミックな振る舞いの変更を実現でき，メッセージ等の外的要因だけでなく内部のトリガーにより必要な振る舞いの実行を実現できる．さらに，複数のモジュールの動作の合理性を管理し，確認できるため，正当性を保持した上での様々な改良／変

表１　シナリオ／ロールの拡張対応表

ロールの記述内容 ＼ シナリオが行なう割り振り	A: オブジェクトとロールの割り振りが一意	B: オブジェクトとロールの割り振りが変化する
(1) 対象システムに特有なプログラム	従来のオブジェクトと変わらない枠組みの変化のみ	振舞いをダイナミックに変更できる
(2) (1)＋振舞いのトリガーや実行制約	動作の関連を確認／管理できる	振舞いを動的に変更可能 動作の関連を確認／管理できる

更の操作を実行できる．この拡張構造を表１に示す．

＜ロールの記述 (1)：部分的プログラム＞

S-R-Oモデルの基本概念である対象領域と共有領域を分けるためには，ロールが，対象システムとしての振る舞いをすべて担当し，シナリオからの要請により，オブジェクトにそれらの付加的な振る舞いを張り付けて利用する形式を実現すればよい．この記法では，従来のオブジェクト指向プログラミングにおけるサブクラスの記述をロールが受け持つことと本質的に代わりがない．この記法では，対象システム側の記述と共有側の記述を分ける枠組みを与えたことがポイントになる．プログラマーがこれらの記述の区別を，正確に行わうことが前提とされる．

＜ロールの記述 (2)：起動トリガーや制約＞

この記法では，対象システムとして必要とする振る舞いを，共有可能なオブジェクトへの差分的な振る舞いの付加での実現に加えて，ロールに振る舞いの起動のトリガーや実行の制約を付随させ，オブジェクトの振る舞いの合理性の管理と確認を行なえるようにする．対象領域側で記述するプログラムには，常にその振る舞いのトリガーや実行時の制約を考慮する．これは，従来のオブジェクト指向プログラミングで，振る舞いの違いをメソッドとして区別していたものを，個々のロールに分け，さらにその振る舞いのトリガーや実行時の制約とともにモジュールとして個別に記述することに対応する．これにより，従来はメッセージの送信側での確認を必要としていた前処理や，振る舞いの起動の指示のためのメッセージは，ロールで管理し，処理される．実質的にはロールがオブジェクトの振る舞いの管理をしていることになる．

＜シナリオでの割り振り A：一意＞

これは，各対象システムごとにオブジェクトとロールの割り振りが一意に決定する場合である．従って，与えられたオブジェクトは，対象システムに必要な振る舞いを実現するために予め決められたロールで静的に改良される．この処理では，ロールの割り振りが静的であることから，実現システムとしては，従来のオブジェクト指向プログラミングのクラス階層による差分プログラミングと基本的に変わらない．しかしながら，必要な割り振りをシナリオがまとめていることから，対象領域と共有領域を分けることは実現される．

＜シナリオでの割り振り B：動的に変化＞

この場合は，ロールの割り振りが動的であることから，ある対象システムで動作するオブジェクトの持っている機能がダイナミックに変化することになる．これは従来のオブジェクト指向プログラミングにおけるクラス／サブクラスの関係を柔軟に変化できるものに対応させられる．ただし，割り振りのダイナミックさを利用するためには，ロールで付加的に記述する振る舞いには単なるサブルーチン的な記述ではなく，様々な状況のもとでの機能の断片を記述することになる．一連の機能は，これらの断片的なロールの組み合わせによって実現する．このようなダイナミックな割り振りによって，対象システムとしてのモジュールは持っている機能を柔軟に変更できるようになる．さらに割り振られるロールの記述内容に応じて，メッセージを受けてその振る舞いを実行するだけでなく，自発的に振る舞いを実行することもできるようになる．

従来のオブジェクト指向プログラミングにおいて，複数のオブジェクトの動きや状態の合理性が捕えられないことの原因は，オブジェクトが実行する動作の関連が理解できないことにあ

る．この根本的問題は，対象システムで行なう一連の動作がメッセージの連鎖で実現され，それらの動作に対して正当性の維持できる切れ目が見えないことである．クラスとして捕えられるモジュールは，対象システムで行なう動作の合理的な切れ目に対応するものではない．これまでに示したようにS-R-Oモデルでは，シナリオ/ロールの拡張を行うことでRoleユニットが，この切れ目を示すことになる．

また，複数のオブジェクト間の動きの連係や状態の合理性が捕えられないことは，オブジェクト間の相互の連帯関係が理解できないことでもあり，プログラミングレベルでは，各オブジェクトが何を継承しているのか判断できないことにもつながる．これは，オブジェクトのモジュールとしての抽象化が，クラス階層という形で実現されているが，対象システムとしての挙動の抽象化の手段が無いことに原因がある．拡張されたS-R-OモデルのRoleユニットは，オブジェクトと独立に動作の連係を起動のトリガーや実行の制約として表すことから，対象システムとしての挙動の抽象化を果たすために重要な概念であることがわかる．

特に，Roleユニットでまとめられている動作に関する情報を元に，それらのRoleユニットの推移構造から，対象システムとしてのダイナミックな動作の関係を静的に確認したうえで，必要な振る舞いの改良／変更などの操作を行なえるようになる．これらの詳細については次の4節で事例とともに示す．

Roleユニットは，システムとしての一連の動作を示すためにセットとして扱われるだけでなく，セット間に新たな構造を持つ．ロールセットの分類の構造は，これらのロールセット間の階層関係や包含関係である．この構造を持ったロールセットは，各ロールセット内では排他的であり，ロールセットの上下関係では上位が下位を包含することになる．

ロールセット間の分類は，リニア構造からツリー構造，ラティス構造，ネットワーク構造と拡張可能である．ロールセットの構造としての排他性や包含関係を守ることで，各構造ごとにRoleユニットに関する推論を実現でき，それらがアサインされるオブジェクトの実際の動作の関係を把握できる情報となる．現在のS-R-Oモデルでは，ロールセット間の分類にはツリー構造を持たせている．このロールセットの構造を段階的に広げていくためには，同じオブジェクトに張り付く複数のRoleユニットの関係を実現するメカニズムを確立する必要がある．

4．S-R-Oモデルの記述事例

ここでは，対象システムの事例として簡単なギアシステムをとりあげ，S-R-Oモデルの記述事例を示す．この事例では，ボトムアップ的に用意されるオブジェクトとしてgearとgear_linkageを用いて，3つのギアが連結したギアシステムの動作を実現する（図2）．

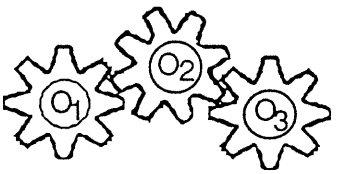

図2　3つのギアによるギアシステム

用意されたObjectユニットは，回転等のギアとしての基本的振る舞いを持ったgearオブジェクトとギアの連結状態や連結により回転が伝播する等の振る舞いを持ったgear_linkageオブジェクトである．これらは，対象領域に依存することなく予め決められる機能を持ったオブジェクトである．

ギアシステムとしての動作のために，必要とされるRoleユニットは，ギアの回転に関するturnロールセット(before_turn, turn, after_turnが要素)である．これらがgearオブジェクトに割り振られる．このロールセットの中のturnロールの記述を図3に示す．turnロールでは，associ-

```
defrole turn ::
 associate
   condition  role(before_turn) ;
     trigger  object is active_message or
              object is remove_role(before_turn) ;
 dissociate
     trigger  object is end(active_message) ;
 active_operation
     refer_type, renewal_type.
```

図3　turnロールの記述

ateによってそれが割り振られるトリガーや条件が示され，dissociateによって剥がれるトリガーが示されている．active_operationは，このロールが割り振られている時に実行できる振る舞いを示すものである．ロール内の記述には，この図のようにオブジェクト側の記述を直接用いない．これが独立性とシナリオによる対応の必要性を意味している．

turnロールセットのロールの推移構造は，各ロールの記述内容から図4のようになる．図からわかるように，このロールセットはbefore_turnロールの割り振りがデフォルトであり，turn_gear/1のメッセージを受け取ることでロールがbefore_turnからturnに変化する．さらに，turn_gear/1の実行を終了した時点で，turnロールが剥がれ，after_turnロールが割り振られる．このロールセットのすべての対象オブジェクトに，after_turnロールが割り振られた状態になった時，before_turnロールに戻ることを示している．

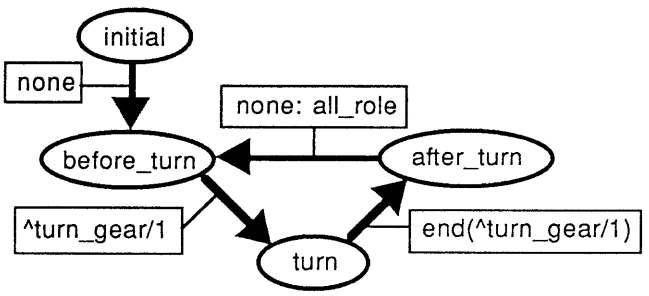

図4　ギアにおけるロールの推移構造図

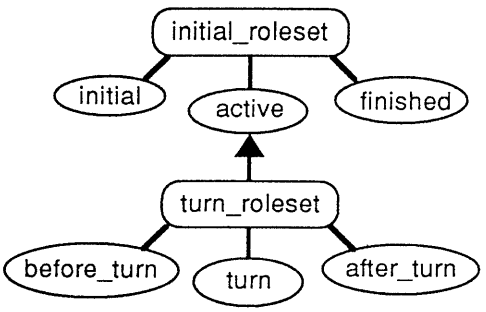

図5　ロールセットの分類構造

また，このロールセットの上位にあるロールセットは，initial, active, finishedの3つのロールからなるinitialロールセットである．これらのロールセットの分類構造を図5に示す．図5の上位／下位関係では，包含関係がある．つまり，turnロールセットの各ロールは，activeロールの性質を受け継いでいる．

5. S-R-O方式によるトランザクション処理法

S-R-Oモデルでは，オブジェクトの動作は，何らかのロールが少なくとも一つ張り付いている上で行なわれる．オブジェクトの動作に関する制約を張り付いているロールが与えることから，オブジェクトの動作を張り付くロールで管理できるようになる．従来のオブジェクト指向プログラミングでは，オブジェクトに基本単位の基準を合わせることしかできなかった．しかし，S-R-Oモデルでは，オブジェクトへのロールの結合がオブジェクトの動作を管理することになり，オブジェクトに横断的な形で，動作の観点からロールのまとまりに対して基本単位を割り当てることができる．

あるシステムにおけるトランザクションを管理するためには，やはりそのシステムにおける基本単位を的確に捕える必要がある[7]．ここで基本単位とは，他の動作から影響や干渉を受けない原子的なまとまりのことである．そして，対象システムとして要求される動作における各基本単位の関わりが明らかであることが重要に

なる．

トランザクション処理のためのS-R-Oモデルの特徴を次のようにまとめられる

○　ロールとオブジェクトの独立性と直交性を規定し，動作の関連についての理解と確認の方策を与えた．

○　ロールが実行時のオブジェクト間のインテグリティを保つ．ロールによって複数のオブジェクトの動作の関係が見える．

○　オブジェクトは共有可能な機能をクラスレベルで明確化し，ボトムアップに用意される

○　シナリオがロールとオブジェクトの割り振りを管理し，ロールとオブジェクトの再利用系を実現する

ここで，他のモデル技術との対応を検討すると，オブジェクト指向概念を基本として，シナリオを中心に設計を進め，プログラミングにつなげる方法は，S-R-Oモデルだけでなくuse-case[8]を始めとしてOBA[9]など幾つかの方法が既に提案されており，実際にツールとして提供されている．しかし，S-R-Oモデルのポイントは，シナリオとして対象システムを捕えるだけでなく，ロールによるオブジェクトの動作の管理や確認が行なえることにある．

6．まとめ

実際に，対象システムとしての動作を把握するためには，個々のオブジェクトが受け持っている動作の内容を理解するだけでなく，オブジェクト間の動作の関わりがどのような要件のもとでどのように変化し，どのように絡み合っているかを理解できる必要がある．これらの情報を理解するための軸がS-R-OモデルにおけるRoleユニットである．S-R-Oモデルでは，対象システムの基本構成要素となるモジュールの理解は，オブジェクトとしての観点とロールとしての観点の２つの直交した軸を用いて，それぞれの観点から行なえる．

重要な点は，オブジェクトの記述に対する再利用性／信頼性が確認でき，同時にオブジェクトの動作に関する関連性／信頼性もチェックできることである．

現段階では，ロールの記述を簡潔にしているため，シナリオが難しくなる点が問題となる．今後は複数のロールの割り振りやロールの記述の再考を進めシナリオの再利用性の検討が必要である．

このS-R-Oモデルでは，ロールが動作のトリガーを持っていることから，ロールが割り振られたオブジェクトは，エージェントとして捕えることが出来る．

参考文献

[1] 片山："多目的意思決定に基づくオブジェクトの組み立て法"日本ソフトウェア科学会第６回大会論文集, pp169-172, 1989

[2] 片山："オブジェクト表現構築のためのクラス構成支援" WOOC '87, 1987

[3] 片山："多視点に基づくオブジェクト指向表現システム" 情処ソフトウェア工学研究会 90-SE-73-5, pp35-42, 1990

[4] 片山, 小林：" ソフトウェアとしての利用性を指向した新しいオブジェクトモデル(S-R-Oモデル) " Proc. Symposium of Information Science 1993, Jan 1993

[5] 片山：" Scenario/Role/Objectモデルによるトランザクション処理メカニズム" 情処第48回大会論文集(5), pp343-344, 1994

[6] Eswaran K.P., Gray J.N.,Lorie R.A, and Traiger I.L. "The Notions of Consistency and Predicate Locks in a Database System" Communications of the ACM, V.19, No.11, 1976, pp624-633

[7] Gray J. "The Transaction Concept: Virtues and Limitations" Proc. 7th Int. Conf. Very Large Database Systems (VLDB) 1981, pp.144-154

[8] Jacobson I. "Object-Oriented Software Engineering " Addison-Wesley, Reading, Mass., 1992

[9] Rubin K. S. & Goldberg A. "Object Behavior Analysis " Communications of the ACM, Vol.35, No.9, 1992