インスタンスの機能拡張が可能な継承モデル

3X－4

鬼塚 真　　小林 伸幸　　小西 史和　　西岡 秀一

NTT 情報通信研究所

1　はじめに

インターネットやイントラネットの普及に伴い，情報の共有化は以前にも増して重要視されてきている．共有した情報を利用する形態は，

- 必要な部分だけを選択して利用する
- 情報に新たな機能を加えて利用する

の2種類に分類可能である．前者については RDB や ODB のビューの概念が適用できる．後者については，ODB における継承の概念が相当するが，現状の継承モデルではスキーマレベルの機能拡張が可能なだけであり，既存のインスタンスに対し新たな機能を追加することはできない．

そこで本稿では，インスタンスレベルの機能拡張をサポートする継承モデルを提案する．更に継承モデルの実装方法について検討する．

本稿の構成を示す．2章では，インスタンスの機能拡張が必要となるアプリケーションである拡張 WWW ブラウザの概要を示し，インスタンスの機能拡張の必要性を論じる．3章では従来技術の問題点を整理する．4章ではインスタンスの機能拡張をサポートする拡張継承モデルを提案し，5章でその実装法を検討する．

2　インスタンス機能の拡張事例 (拡張 WWW ブラウザ)

インスタンスの機能拡張を利用することにより，URL の特徴毎に異なる振舞いを定義できる WWW ブラウザを実現することができる．本ブラウザの永続オブジェクトモデルの概要を図1に示す．

図1での主要なクラスは URL 自身を表す *Url* クラスと URL 間の関係を表す *UrlRelation* クラスである．*PRoot* クラスはこれらのクラスのインスタンス群を集約し管理するためのクラスである．*Url* インスタンスは自分自身の URL 情報を元にブラウズ画面を表示 (*open*()) 可能であるが，サーチエンジンの URL ではこれとは別に利用者の検索履歴も表示できるようにするため *Url* クラスを拡張した *SearchUrl* クラスを定義している．もしインスタンスの機能拡張がサポートされているならば，WWW ブラウザの利用者は *Url* インスタンスを選択的に *SearchUrl* クラスに移動することによって，*open*() により *SearchUrl* クラスの *open*() が起動され，利用者検索履歴を表示することが可能となる．

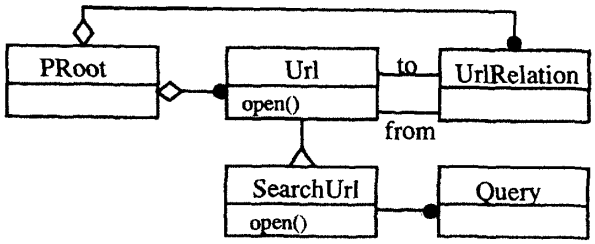

図 1: 拡張 WWW ブラウザの永続オブジェクトモデル

3　従来技術の問題点

現状の ORDBMS, ODBMS を用いて，前章で説明したようにインスタンスの機能を拡張するには主に以下の2つの手段が考えられる．

インスタンスの削除と生成　図1の *Url* インスタンスを削除し，*SearchUrl* インスタンスを新規に生成する方法であるが，インスタンスの OID が変化するため *PRoot* インスタンスとの間の参照一貫性を保障できない．

委譲によるインスタンスの機能拡張　インスタンスの機能を拡張するために *Url* クラスと *SearchUrl* クラスの間に継承を用いずに委譲 (文献 [1]) を用いる方法である．この場合，動的束縛が利用できないため，*Url* インスタンスに対する *open*() が *Url* クラスに対するものか *SearchUrl* クラスに対するものかを，アプリケーション側で実装しなくてはならない．

Inheritance Model to Support Instance Extension

Makoto ONIZUKA, Nobuyuki KOBAYASHI, Fumikazu KONISHI and Shuichi NISHIOKA

NTT Information and Communication Systems Laboratories

4　拡張継承モデル

インスタンスの機能拡張を実現する手段として，本稿ではインスタンスのクラス移動とインスタンスレベルの共有の概念を提案する．

インスタンス移動 OID を変更せずに指定のインスタンスを下位クラスのインスタンスに移動する．本機能により，参照一貫性を保障したまま指定のインスタンスの機能を拡張することが可能となる．

インスタンス共有 下位クラスのインスタンスを新規に生成し，生成したインスタンスの属性値として指定したインスタンスの属性値を参照する．本機能により，指定のインスタンスの機能を共有したインスタンスを生成することが可能となる．

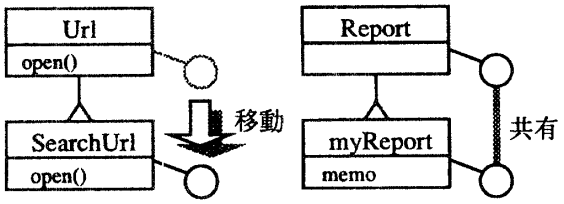

図 2: インスタンス移動とインスタンス共有

4.1　追加した構成子

継承リンク インスタンス共有を表現するために用いる．インスタンス間の半順序関係であり，インスタンス間に継承リンクが定義されている場合，上位のインスタンスの属性値の集合が下位インスタンスと共有されることを意味する．

4.2　追加した操作

移動 (*Instance*, *fromClass*, *toClass*) *Instance* インスタンスを *fromClass* クラスから *toClass* クラスへ移動する操作である．但し *Instance* インスタンスの生成時の型は *fromClass* クラスであり，且つクラス上の継承関係において *fromClass* クラスは *toClass* クラスの上位であることとする．

共有生成 (*Class*, *superInstance*) *Class* クラスのインスタンスを生成する際に *superInstance* インスタンスとの間に継承リンクを設定する操作である．但しクラス上の継承関係において *superInstance* インスタンスの生成時の型は *Class* クラスの上位であることとする．

5　拡張継承モデルの実装法

拡張継承モデルにおける永続インスタンスの格納方法として文献 [2] の垂直分割を採用する．本方式により，同文献の水平分割と比較してインスタンス移動，インスタンス共有の際の処理量が軽減できる．垂直分割による格納構造を図 3 に示す．

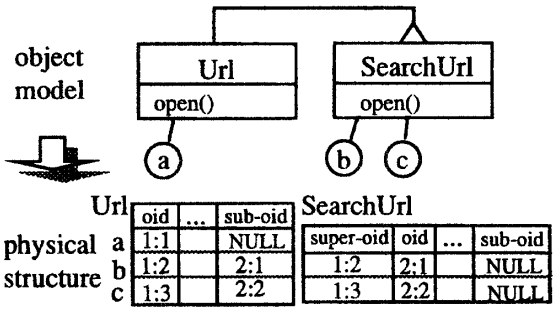

図 3: 垂直分割による永続インスタンスの格納

インスタンス移動を実現するには，図 3 の垂直分割の格納構造がそのまま利用できる．例えば，図 3 において移動 ($a, Url, UrlSearch$) を実施した場合，*UrlSearch* クラスのインスタンスを新規に生成し，そのインスタンスの OID を a の $sub-oid$ に設定する．そして生成したインスタンスの $super-oid$ に a の OID を設定する．

一方インスタンス共有を実現するには，下位のインスタンスの OID を複数格納するため図 3 の格納構造に $sub-oid-list$ を追加する．共有生成 (*UrlSearch*, a) を実施した場合，*UrlSearch* クラスのインスタンスを新規に生成し，そのインスタンスの OID を a の $sub-oid-list$ に追加する．そして生成したインスタンスの $super-oid$ に a の OID を設定する．

6　おわりに

今後は，我々が現在開発している ORDBMS に対し本稿で提案した継承モデルを導入し，拡張 WWW ブラウザに対しての有効性を確認する予定である．また垂直分割による永続インスタンスの格納方式の性能についても検討していきたい．

References

[1] J.Rumbaugh et al, "Object-oriented modeling and design," Prentice-Hall, New York, 1991.

[2] U.Hohenstein, "Bridging the gap between C++ and relational databases," ECOOP'96, pp.398-420, 1996.