分散環境に適した情報検索手法の検討

3T－6

脇田 竜一　東京大学
浜田 喬　学術情報センター

1 はじめに

最近のコンピュータネットワークの急速な規模拡大により，各種のデータベース，知識ベース，電子図書館，ファイルシステムといった様々な情報ベースが広く利用されるようになっている。これに伴い，複数の情報ベースを統合的に利用したいという要求が高まりつつある。

しかしこれら情報ベースのシステムの多くは旧来のユーザインターフェイスを継承しており，必ずしもそのままでは協調的利用に適さない。また本来それぞれが独立に設計，構築されているこれらのシステムの間でスキーマや情報構造の統一は望むべくもなく，複数の情報ベースから得られた情報を矛盾なく統合された形でユーザに提示するには，検索の後処理としてなんらかの情報加工が必要である。

本稿では異種分散環境に対応した検索システムの作成を目的として，version の概念を導入することにより統一されたオブジェクトビューを得るための手法を提案する。検索システムの基本的構成，version の概念と実現法，および検討と今後の課題について述べる。

2 分散環境での検索手順

文献 [1] では広域検索のための問い合わせ手法と結果の統合のための基礎的なシステム構成が提示されている。これによると統合化検索システムのモジュールとして (1) 統合検索のユーザインターフェイス，(2) 個々の情報ベース固有の形式へ問い合わせや結果を変換する情報ベースインターフェイス，(3) 各情報ベース間の情報不整合を解決するための制約解消モジュール，(4) 非構造化情報ベースから検索に必要な構造情報を取り出す構造抽出モジュール，などが必要とされている。

中でもユーザインターフェイスは統合情報検索環境をユーザに提供する上で重要な役割を担うシステム要素であり，次のような処理が要求される (図 1)。

- ユーザの問い合わせを解析し，問い合わせるべき情報ベースを決定する。
- 問い合わせ先が複数の情報ベースにわたる場合，ユーザの問い合わせを適切に分割した上で，各情報ベースへ問い合わせを発送する。
- 複数の情報ベースから返ってきた結果の間の重複・矛盾や情報構造の違いを整合化する。
- 複数の情報ベースから返ってきた結果をユーザに提示しやすい形に統合化する。

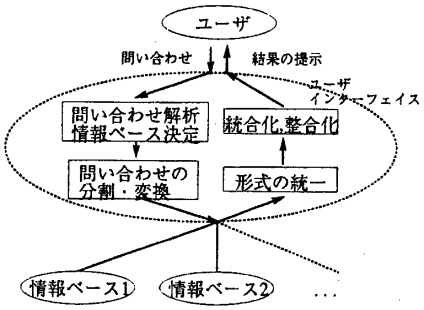

図 1: 分散問い合わせモデル

3 Version を用いたオブジェクト表現

前節で述べたユーザインターフェイスの 4 段階の処理は，それぞれを孤立したサブモジュールによって構成することも可能である。しかし，ユーザから見たデータ表現を自然に反映させるには，オブジェクト表現によるカプセル化が有効である。

オブジェクトモデルの差異や変化に動的に対応する手法として，version によるオブジェクトの管理がある。本稿では multi-methods を念頭に置き，次のようなモデルを提案する。

- 単一のオブジェクト実体に対して複数のメソッド実装を許し，実装ごとに異なる version id を与える。
- ユーザは version を区別せず，generic な名前のみを指定してオブジェクトにアクセスする。
- 実際にどの version が使用されるかは，オブジェクト使用状況に依存する。これを version 選択の規則としてあらかじめ記述しておく。

ここではリストによってオブジェクト間のバージョン依存関係を示す方法を考える。依存規則リストは一般に次のように書ける。

```
(Class1(C1_ver1(Class2 C2_ver1)(Class3 C2_ver1)..)
        (C1_ver2(Class2 C2_ver1)..)
  ...
)
```

このリストの一行目は，「もし Class1 オブジェクト

An Information Retrieval Method for Distributed Environments

Ryuichi Wakita[1], Takashi Hamada[2]
[1]University of Tokyo
[2]National Center for Science Information Systems

の version が C1_ver1 ならば， Class2 オブジェクトの version は C2_ver1 とし，Class3 オブジェクトの version は C3_ver1 とする」という意味である。規則を再帰的に辿ることにより，他のオブジェクトの version に依存するオブジェクトの version を順次決定できる。

version が単なるサブクラスと異なるのは，オブジェクトアクセス時に規則によって自動的に version が決定されるため，ユーザからは version を意識せずにアクセスできるという点である。すなわち，実際にどの version が使用されているかに関わらず単一のクラスオブジェクトとして振舞っているように見せることが可能である (図 2)。

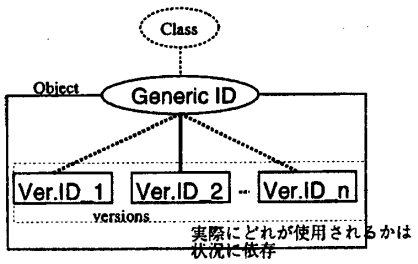

図 2: version によるオブジェクトビュー

4 問い合わせ機構の version 化の例

前節に述べた version 機構をもとに，複数の情報ベースにわたるオブジェクトビューを構築する例を示す。

コンピュータセットの購入見積りを求める場合を考える。発注可能な取引先が 2 つあり，全部で 7 つのデータベースが存在すると考える。

course1: (クラス: Course) 取引先 1 で販売しているセットの組合せ

cpulist1: (クラス: Cpu) 取引先 1 の cpu 製品リスト

disklist1: (クラス: Disk) 取引先 1 のディスク製品リスト

course2: (クラス: Course) 取引先 2 で販売しているセットの組合せ

cpulist2: (クラス: Cpu) 取引先 2 の cpu 製品リスト

disklist2: (クラス: Disk) 取引先 2 のディスク製品リスト

supplylist2: (クラス: Supply) 取引先 2 のサプライ製品リスト

予算が 50 万円以下で，CPU が Cpu1 のセットの見積りを求めるために次のような問い合わせを実行したい。

```
select course.name
from course in Course
where course.cpu=Cpu1 and course.total()<=500000
```

ここで，where 節に現れるメソッド course.total() は Course オブジェクト内で参照されている Cpu オブジェクト，Disk オブジェクトおよび Supply オブジェクトの価格の総和を求めるメソッドである。これは複数のデータベースにまたがる操作であり，特定のデータベースの内部では実行できず，ユーザインターフェイスで提供される。

このメソッドを実行するには，ユーザインターフェイスモジュールが現在参照しているオブジェクトに応じて，同一クラスに属する情報ベースからアクセスすべき情報ベースを一つ選択しなければならない。このための規則を依存リストに記述することになる。上例の場合，依存リストは次のようになる。

```
(Course  (Course1 (Cpu Cpulist1) (Disk Disklist1))
         (Course2 (Cpu Cpulist2) (Disk Disklist2)
         (Supply Supplylist2))
        )
```

version によって複数の情報ベースを併合したオブジェクトビューが得られ，並立する情報ベースのどれにアクセスするかの選択を依存リストをもとにして動的に決定できる (図 3)。

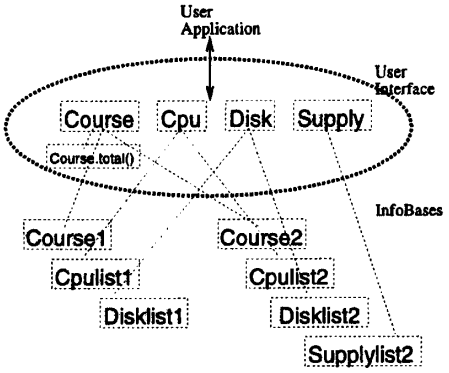

図 3: 統一オブジェクトビュー

5 おわりに

本稿では，分散環境中に存在する複数の情報ベースから抽出した情報に対し，オブジェクトの version の概念を利用することによって統一的なオブジェクトビューを与える手法を提案し，複数の情報ベースへのアクセスと情報の統合を同時に管理する方法について簡単な例に即して述べた。ただし，分散環境検索システムとして実現するにはまだ不十分な点が多い。課題としては (1) 問い合わせの分割処理，(2) 依存関係情報の抽出と維持，(3) 制約の管理，などが挙げられる。

以上の課題を含め，今後システムプロトタイプの作成と評価を通じて順次具体的に検討していく予定である。

参考文献

[1] S.Chawathe,et.al.:"The TSIMMIS project:Integration of Heterogeneous Information Sources," 情処研報, DBS-100-2, Oct.1994.