動的クラス定義可能なディレクトリサーバの実装と性能評価

3V－5

古賀靖人　安村義孝
NEC C&C メディア研究所

1　はじめに

ディレクトリサービスの標準プロトコルとしてLDAP (Lightweight Directory Access Protocol)[1] が普及するのに伴い、ディレクトリサービスの利用が様々な場面に拡大してきている。ディレクトリサーバはエントリと呼ばれるデータ単位の階層を記憶している。各エントリはそれが何を表現しているかを示すオブジェクトクラスを持ち、オブジェクトクラスに応じた属性を持っている。

ディレクトリサーバはエントリの階層を格納する必要があるため、実装にはオブジェクト指向データベースが適していると考えられる。しかし、各属性値を個々のオブジェクトとして格納すると、属性を参照するコストが増大する可能性がある。一方、エントリをオブジェクトクラスごとに定義したクラスで表現し、属性をクラスのメンバとする実装も考えられる。この実装では属性の参照コストは小さくなるが、属性値の数が1個もしくは有限個に制限されるためサーバの汎用性が失われてしまう。

本論文では、オブジェクト指向データベースの実行時インターフェースを用いて動的にクラス定義を行なうことを考える。これにより、用途に応じて適切なクラス定義を行うことで、サーバをコンパイルすることなく汎用的な実装よりも高い性能を実現できると考えられる。

この方法の有効性を検証するため、実行時にクラス定義を行ない属性をメンバとして格納するLDAPサーバと、各属性値を個々のオブジェクトとして格納するLDAPサーバを、オブジェクト指向データベースPERCIO[2] 上に実装し、実験によって性能を比較した。実験の結果、検索リクエストの処理において約10% の性能向上を確認できた。

2　設計と実装

2.1　動的クラス定義版サーバ

動的クラス定義版サーバは、用途に応じてエントリの実装を変更可能にするため、属性に関して2種類の実装の多様性を提供する。1つは個々の属性値の実装であり、例えばcharの固定長配列やintなどとして実装することが考えられる。もう1つは属性の実装であり、1つの属性値のみを持つ実装や、任意個数の属性値を集合として持つ実装などが考えられる。

この多様性を提供するため、属性値の基底クラスTEAttrValと属性の基底クラスTEAttrを定義した。TEAttrValはLDAPのインターフェースから渡された値と格納される表現との間の変換を行なう仮想関数を持ち、TEAttrは、属性値の追加や削除を行なう関数、属性値に対する反復子を返す関数などを仮想関数として持っている。これらの仮想関数を実装した派生クラスを用意しておくことで多様な実装を提供する。

現在の実装では、charの固定長配列とintとして実装した属性値のクラスと、それを1つの値として持つ属性のクラス、およびそれらの任意個数の集合として実装した属性のクラスをあらかじめ用意してある。また自分で属性値や属性を表現するクラスを定義したい場合は、コンパイルが必要になるが、TEAttrVal、TEAttrを派生させることにより容易に実現できる。

エントリを表すクラスの定義は、オブジェクトクラスをサーバに登録する時に使用する属性とそれに用いるクラスを指定して行われる。現在の実装では、これはサーバがデータベースを初期化する時点でのみ行われる。

2.2　汎用版サーバ

汎用版サーバは、各属性値を個々のオブジェクトとして格納する。エントリは、オブジェクトクラスによる制限を除いて、任意の属性に対して任意個数の属性値を持つことができる。

エントリTEntryは属性をTEntryAttrの集合として保持し、TEntryAttrは属性のタイプと、属性値およびその長さを保持している。属性値は動的に確保した可変長のchar配列に格納される。

3　実験

3.1　実験内容

使用したディレクトリデータは、3段の組織単位 (1段目から順に8, 64, 512 エントリ) と4段目に人 (10240 エントリ) という合計10824 エントリの階層である。組織単位のエントリが持つ属性は、オブジェクトクラス

Directory Server with Dynamic Class Definition
Nobuhito Koga and Yoshitaka Yasumura
C&C Media Research Laboratories, NEC Corporation

(objectclass)、組織単位(ou)であり、人のエントリが持つ属性は、オブジェクトクラス(objectclass)、氏名(cn)、姓(sn)、メールアドレス(mail)、電話番号(telephoneNumber)となっている。

動的クラス定義版サーバにおけるエントリのクラスは、このディレクトリデータにほぼ合わせて定義した。具体的には、人エントリのクラスはcn, sn, mail, telephoneNumberをそれぞれchar[64]として1つの属性値を持ち、組織単位エントリのクラスはouのほか9個の属性をそれぞれchar[64]として1つの属性値を持つように定義した。

実験用マシンはCPU Pentium 200MHz、メモリ96MBのNEC PC-9821St20で、OSはWindows NT4.0 Serverを用いている。サーバとクライアントプロセスは同じマシン上で実行させた。

クライアントはフィルタとしてcnを指定した検索を30000回行ない、それが完了するまでの所要時間を計測した。指定するcnの順序はランダムに選んである。得られた値から、1秒間あたりに処理されたリクエストの数を求めた。計測は6回行なって、最も時間の短かった結果を採用した。これは、OSがバックグラウンドで行なう作業の影響などを避けるためである。

なお今回の実験では、2種類のサーバとも1分間隔でcommitを行なうようになっている(オプションによりリクエストごとに行なうことも可能)。本来は更新リクエストの処理後にcommitする必要があるが、今回の実験では検索しか行なわないので問題ない。また、実使用においても読み出しがほとんどだと言われていることから、実使用における性能と大きく異なることもない。

3.2 結果と考察

エントリ検索の測定結果を図1に示す。結果を見ると、動的クラス定義版サーバが約10% の性能向上を示している。これは、属性をクラスのメンバとして実装することにより、属性の参照コストが小さくなっているためである。

検索の性能差が約10%とやや小幅にとどまった原因はいくつか挙げられる。まず、リクエストの処理にはインデックスを用いた検索処理なども含まれ、属性値の参照コストが必ずしも全コストの大部分を占めているわけではない。

また、実験用のデータがやや小さめであることからデータがメモリにのっている場合が多く、汎用版の性能の劣化があまり起きていないことが挙げられる。さらに、本稿における実験では全てのエントリを追加した直後の状態で検索の性能を測定しているため、汎用版サーバにおいても同じエントリの属性値はディスク上にまとまって格納されている。実際の使用においては、データの更新が行なわれて同じエントリの属性値がディスク上の離れた位置に格納されると、性能が劣化する。従って、データがより大きく更新がある程度行なわれた後の、実使用に近い状態では、動的クラス定義版サーバによる性能向上はより大きくなると予測できる。

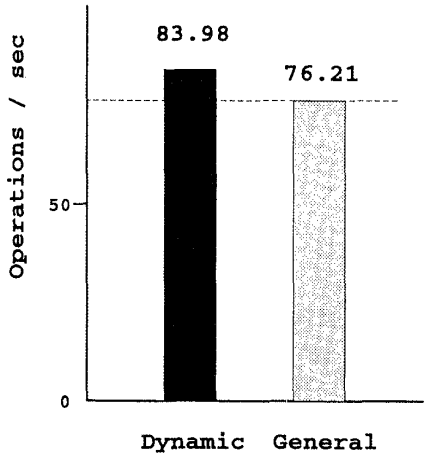

図1: エントリ検索の測定結果

4 結論

本論文では、実行時にクラス定義を行ない属性をクラスのメンバとして格納する動的クラス定義版ディレクトリサーバを実装し、各属性値を個々のオブジェクトとして格納する汎用版サーバと性能を比較する実験を行なった。その結果、検索リクエストの処理において約10%の性能向上を確認し、この方法の有効性を示した。

また性能以外の点でも、属性値をクラスのメンバとすることにより、selectが使用できるなど、LDAPのインターフェースからだけでなくデータベースに直接アクセスするのが容易になるという利点もある。

今回は実装を単純にするために属性と属性値の基底クラスを用意してその派生クラスを定義することにより多様な実装を提供したが、この方法では多くの属性が複数の属性値を持つ場合には効率が悪い。複数の属性値を持つ属性は汎用版サーバのような実装にすることで、この効率の悪化を避けることができる。

参考文献

[1] M. Wahl, T. Howes, and S. Kille. Lightweight directory access protocol (v3), 1997. RFC-2251.

[2] 鶴岡 邦敏、木村 裕、波内 みさ、安村 義孝. オブジェクト指向データベース管理システムPERCIOの開発と今後の課題. 電子情報通信学会論文誌, Vol.J79-D-I(No.10):pp.587–596, 1996.