マルチポイント ・ニュースリーダの実装と評価

Multi-server Multi-point News System

島袋 政辰[†]　　河野 真治[††]
Masatatsu SHIMABUKURO　　Shinji KONO

[†]琉球大学理工学研究科情報工学専攻
Specialty of Information Engineering, University of the Ryukyus
[††]琉球大学工学部情報工学科
Faculty of Information Engineering, University of the Ryukyus

概 要

マルチサーバ ・ニュースリーダは、複数のサーバからニュースを整合性良く取得するものであり、マルチポイントニュースサーバ は、一人のユーザが複数の計算機からでも rlogin することなく一貫してニュースを読むことを可能にするものである。これらに対応するような、新しいニュースリーダの開発について述べる。このニュースリーダは history データベースだけを保持する中間的なサーバを利用する。

本論文ではデータベースの構成、サーバの実装と評価について述べる。

1　はじめに

ネットニュースは、ニュースサーバが配布しているいろいろな情報をユーザが受け取りそれに対しての議論をユーザの間で行われるシステムである。

ユーザは今では複数の計算機を使用するのが普通であり、決まった計算機からニュースを読み出すとは限らない。また、現在では情報の多種多様化に伴い多くのニュースサーバが立ち上げられている。そこで、複数の計算機から remote login することなく一貫してニュースを読むことができるマルチポイント ・ニュースリーダや、複数のニュースサーバから記事の重複や混乱なくニュースを読むことのできるマルチサーバ ・ニュースリーダが必要であると考えられる。

今のインターネットの世界では、複数のニュースサーバに対して複数の計算機 (ユーザ) に対応しているようなリーダが望ましい。本研究ではそのような複数のニュースサーバと、複数の計算機上のリーダの間に立つ中間的なサーバを導入する。中間サーバは、ただ中継をするだけではなく記事に関する情報だけを保存し、必要に応じて複数のニュースサーバに接続する。

そのサーバによりユーザが複数のサーバに接続する手間が省けることになる。

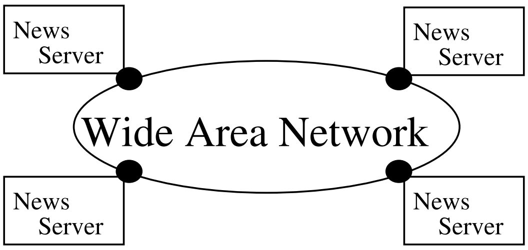

図１　ネットニュースの全体像

2　ネットニュースの仕組み

2.1　ネットニュースとは

ネットニュースはインターネット上で世界中のユーザが公開された形で議論を行うためのシステムである。

複数のニュースサーバがニュースを交換しあい、

大きな論理的なネットワークを構成する。そのネットワークを一括して管理しているサーバはない。(図1)

ニュースシステムを運用している計算機のことをニュースサーバといい、ユーザはそのニュースサーバが配布しているニュースグループを購読して参加することができる。

投稿されたニュースは、ニュースサーバ間でメッセージIDやニュースの本文を送受信する。ニュースサーバが持っているデータベースのhistoryファイルにそのニュースの保存期間を保存し、spoolというディレクトリにニュースグループファイル(ディレクトリ)を作ってそこに本文を保存する。その保存されたニュースをユーザに配布するシステムになっている。(例、news.testの記事はspool/news/test/記事番号で保存)

ニュースシステムとユーザの間での記事の送受信は色々あるが、現在ではネットワークを使ったNNTP(Network News Transfer Protocol)が主流となっている。ニュースリーダがLISTを使って全てのニュースグループに関する情報(モード、古い記事番号、新しい記事番号)を受け取り、GROUPで1つのニュースグループに関する情報(古い記事番号、新しい記事番号、記事の量)を受けてこのニュースグループに入り、ARTICLE(HEAD、BODY)によって記事を受けて読むことが出来る。(図2) NNTPはニュースリーダによってニュースを読む際にそのニュースに関する情報をやりとりする時に用いられる通信規約である。

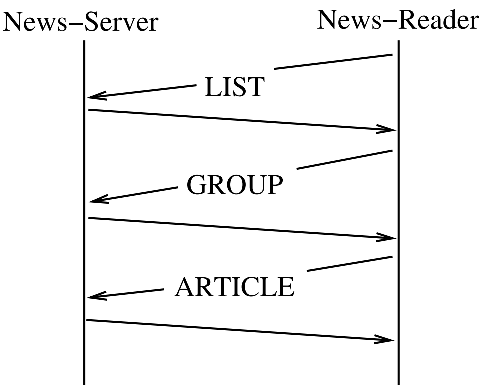

図2 NNTP

2.2 INNの仕組み

ネット・ニュースシステムの代表的なものに、Cnews、Bnews、INNというのがある。本研究ではINNというニュースシステムを利用しているのでINNについて説明をする。

INN自身はニュースを送ることはしないのでニュースを送り出すようにその他のプログラムに指示をだす。実際の配送はnntpsendによって行われている。これらの設定はnewsfeeds、nntpsend.ctl、hosts.nntp、expire.ctlファイルにで設定される。newsfeedsには配送するニュースグループ、パス名を設定し、nntpsend.ctlに配送する計算機のフルドメイン名を設定する。それに従ってnntpsendはニュースを配送するのである。hosts.nntpは配送するニュースサーバのフルドメイン名を設定し、ニュースを受信する。expire.ctlはニュースの保存期間を設定する。

また、ニュースリーダから接続された場合、inndはnnrpdプロセスを生成(fork)し、接続をnnrpdに渡し、innd自体はニュースの配送に集中するのである。

3 マルチポイント・マルチサーバ

3.1 従来のニュースサーバとニュースリーダ

従来のニュースサーバでは、keyがメッセージIDで、あと記事を保持する期間、ニュースグループ名/ニュースナンバーを保持しているデータベース(history)を持っている。他のニュースサーバから送られたニュースはhistoryデータベースにメッセージIDがないか調べ、無ければspoolにニュースを保存するようになっている。(2章を参照)

従来のニュースリーダは、個人の既読情報を.newsrcに書き込んでいる。. newsrcにはニュースグループと購読、ニュースの読んだ範囲が保存されている。(例えば、ura.testを購読し4から56まで読んだとすると.newsrcにはura.test: 4-56、ura.testを購読してないならura.test!となる。) NNTP(2)を通してニュースを取り込むのだが、LISTを使ってニュースグループが持っているニュースの範囲を取り出し、.newsrcと比べ未読既読の情報をつくり出す。(例えば、LISTではura.test 0000000060 0000000001 y、.newsrcにはura.test: 4-56だとすると未読は1-3、57-60となり7となる。)

3.2 マルチポイント・マルチサーバ

3.2.1 データベースの構造

マルチサーバでは、個々のニュース本文の情報をニュースサーバから取るためにメッセージIDをデータベースとして持っておく。そのメッセージIDによってニュース本文をニュースサーバからニュースリーダへ送るようになっている。また、ニュース本文を必要としないプロトコル(LIST、GROUP、LISTGROUPなど)に対しては、ニュースサーバに接続せずデータベースを使って対応するようにしている。そのためデータベースとしては、メッセージIDをkeyとしてメッセージID、ニュースグループ名/ナンバーが保存されているデータベース、メッセージID、ニュースサーバ名が保存されているデータベースの2つを持っている。(図 ??)

マルチポイントでは、アカウントを持っているどの計算機からニュースを読むとしても個人情報(未読、既読)は変わらない。しかし、今のNNTPに対応している既存のニュースリーダ(mnewsやgnuなど)は.newsrcから個人情報を手に入れているため既存のニュースリーダは使えない。個人情報を保存に当たっては今のところメッセージIDをkeyとして読んだ記事にはメッセージIDとユーザ名(ID)を持つデータベースを考えている。まずニュースグループとナンバーでメッセージIDを探し、そのメッセージIDからユーザ名(ID)を探す。データベースにユーザ名(ID)があれば既読で無ければ未読となる。(図 ??)

3.2.2 プロトコル

ニュースサーバとユーザからの接続に対するマルチポイント・マルチサーバの動きをプロトコルにし、そのプロトコルを図 ??に示す。

ユーザ側からの命令に対してはspoolに関する情報(ARTICLE、BODY、HEAD)以外はマルチサーバだけで行ない、ニュースに関する情報は、そのニュースを保持しているニュースサーバから情報を入手して、そのままユーザに流すといった流れになっている。

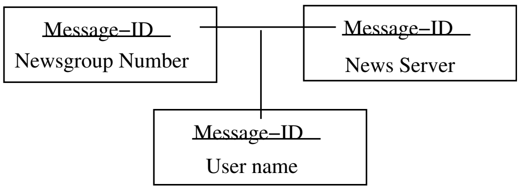

図3 データベースの構造

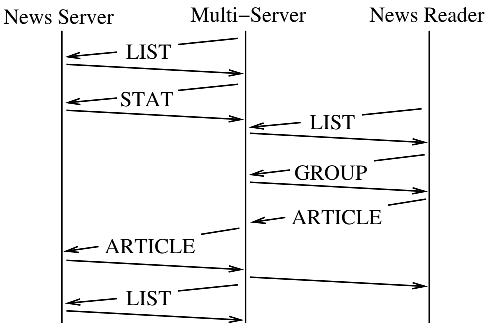

図4 マルチポイント・マルチサーバのプロトコル

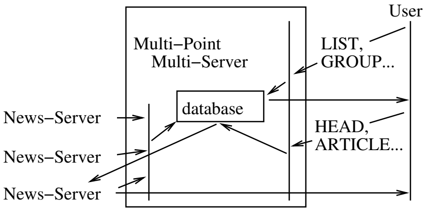

図5 プログラム全体像

4 マルチスレッドによるマルチサーバ

マルチスレッドによって、1つのスレッドはデータベースを作成し、別のスレッドでユーザからのアクセスに答える。また、データベースを作成するスレッドでは、1つ1つのニュースサーバに対応するスレッドが作成され、並列にデータを取り込むようになっている。(図 ??)

1つのスレッドについての動作は、ニュースサーバにLIST(NNTP)を送ってニュースグループのデータを受け取り、それに従って1つ1つのメッセージIDをndbmを使ってデータベースに保存する。(図 ??)

5 マルチポイントの実装について

マルチポイントについては、実際に使っているニュースリーダ(mnews)を改良して実装した。

mnewsとは、NNTPを通じてニュースを読むことができ、LISTを使ってニュースサーバが保持しているニュースグループとそのニュースグループに投稿されているニュースのリストを取り出す。そのリストとユーザのホームにある. newsrcを比べて未読、既読情報をユーザに提供する。それをもとに

ユーザがニュースグループを選んで、それに対応するニュースを提供するようになっている。

ニュースリーダを起動すると MODE READER にせず、マルチポイントの MODE にして従来のニュースリーダと区別する。その MODE では従来のプロトコルとは違ったマルチポイント用のプロトコルを使用する。ニュースリーダからユーザ名 (ID) を送って、LIST と未読記事の量を受け取り、ユーザに提供する。mnews はアルファベット順にニュースグループを提供しているため、マルチポイントでも同じようにニュースを提供するようにする。例えば、test と misc というニュースグループでは、ニュースリーダの起動時に misc の HEAD を先読みしユーザが misc からニュースを読めば時間がかからない。それで misc の記事を読んでいる時に test の HEAD をキャッシュとして持っておく。(図 ??)

そうすることによって、データベースの検索にかかる時間やネットワークの遅さを隠すことが出来る。ニュースの未読既読に関しては 3.2.1 章で述べたようにメッセージ ID をとユーザ名 (ID) のデータベースによって保存する。

6 実装に対する評価

6.1 保持しているデータ量

研究室にあるニュースサーバとマルチサーバのデータベースのデータ量の差を比べる。(表 ??)

マルチサーバの方のデータ量が減っているのが分かる。マルチポイントのデータ量 (2M) を加えてもおよそ 6M である。従来のニュースサーバに比べると spool が無い分データ量も少ないのだが、メッセージ ID 自体が大きいので 3 つのデータベースに key として保存するよりメッセージ ID をローカルな ID として保持するとそのデータ量をもっと減らすことが可能だと考えられる。

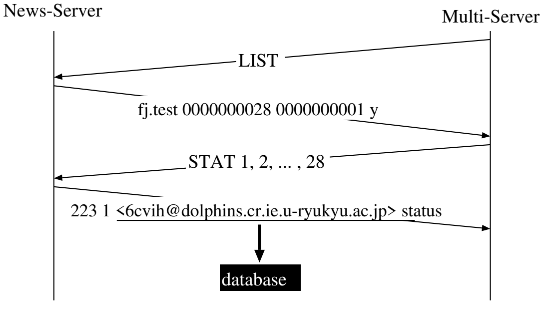

図 6　データベースの作成

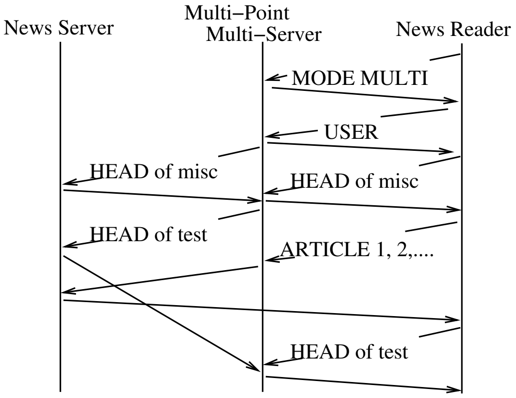

図 7　マルチポイント

サーバ	データ量
ニュースサーバ (history と spool の合計)	33719114(byte)
マルチサーバ	3850240(byte)

表 1　データ量の差

7 まとめ

本研究では、資源の確保やユーザ インターフェースを考えた 「マルチポイント ・マルチサーバ」を導入した。

また、ユーザの立場においても読みたいニュースのためにニュースサーバを変えて接続する必要もなくなった。

参考文献

[1] 安室 貴史, “動的木構造を用いたニュース ・リーダの開発,” July 3 1995.

[2] 山口 和紀, 干 旭, 中村 敦司, 新城 靖, 西山 博泰, 古瀬, 一隆, 石川 佳治, 佐々木 重雄, 林 謙一, 荻原 一隆, 金谷 英信, 鈴木 孝幸, 黒石 和浩, “The UNIX Super Text. 技術評論社” 1992.

[3] Rich Salz, “Inter Net News: Usenet transport for Internet sites,” 1992.