


統一インタフェースをもったネットワーククライアントツール

2T-4

天海良治 奥乃博

 NTT ソフトウェア研究所

1. ネットワーク分散環境

ネットワークを利用したサーバクライアントモデルにもとづいたソフトウェアは、共有による資源の効率的な利用と情報の一元管理、負荷分散による処理能力の向上、ネットワーク透過性を高めることによる操作性の向上など、計算機群全体の機能向上を主目的としている。

だが、ネットワーククライアントを実現することは、有用なソフトウェア資産を容易に利用する手段としても有効である。つまり、異種 OS の下でもクライアントを実現するだけで、サーバ側のソフトウェアを移植せずに利用することが可能となる。さらに、同じプロトコルのサーバがいくつかの機種で実現されている場合、機種の差違はプロトコルのレベルで吸収することができる。また、このプロトコルレベルでの対話性は比較的 low レベルであるので、利用者インタフェースを別に用意する必要があるが、逆にこのことを利用して、各種のソフトウェアの利用者インタフェースの統一を図ることも可能である。

2. LISP マシンでのネットワーククライアント

我々は、LISP マシン TAO/ELIS にいくつかのネットワーククライアントソフトウェアを実現して、UNIX のソフトウェア資産の活用を目指した。また、実現にあたっては利用者インタフェースの部分に TAO/ELIS の画面エディタ ZEN (Zeroth Editor for NUE) を用い、インタフェースの統一を図った。

2.1 ネットワーク部の実現

サーバと通信をするネットワーク部は TAO/ELIS に実装されている TCP/IP ネットワークでサポートしている TCP-stream を使用した。TAO/ELIS のネットワークはオブジェクト指向で書かれており、TCP-stream は TAO の基本的入出力ストリームである fundamental-stream を継承している。これにより、TAO の入出力関数で文字ベースの通信を簡単に行なうことができる。

2.2 利用者インタフェース部の実現

利用者インタフェース部には画面エディタのバッファを利用した。ZEN エディタもオブジェクト指向で実現されており、編集用のバッファはオブジェクトとなっている。インタフェースは、エディタバッファとその操作のメソッド、キーボード入力とメソッドをマップするコマンドテーブルなどから構成されている。このバッファのクラスはエディタの基本バッファのサブクラスとして新たに定義する。これにより、基本バッファに備わっている編集コマンドのコードがそのまま利用できること、ツール用のコマンド入力方法が日常的に利用しているエディタと統一がとれること、日本語の入力や探索も同一のものが利用できること、ファイルの扱いをエディタに任せることができること、機能拡張がコマンド単位で容易にできること、個人の好みにあわせたカスタム化が容易なこと、などの多くの利点が少ない実現の手間で得られた。

2.3 分散環境の問題点

(1) 文字コードの問題

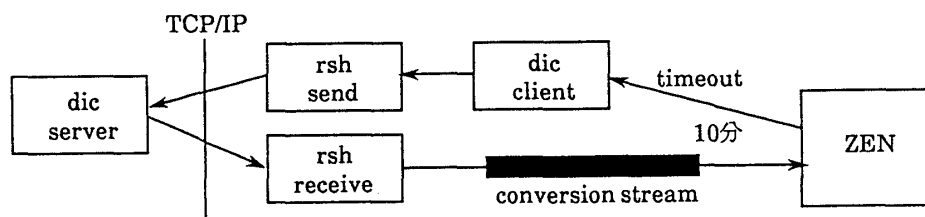
ニュースの記事の部分などには、JIS コードによる 2 バイト文字が多く使われている。TAO での 2 バイト文字は EUC を採用しているため、通信にあたっては文字コードの変換が必要である。また、テキストの行末の印に CRLF の 2 バイトを使用しているため、これも合わせる必要がある。変換には文字コードの変換や行末の調整の機能を備えた入出力ストリーム (conversion-stream) を利用した。これは、他の任意の入出力ストリームを接続して、そのストリームに流れるデータのコードの変換を行なうパイプとして利用できるものである。これも fundamental-stream を継承しているため一般の入出力関数が使える。コード変換部の本体はマイクロコードで書かれているため十分に高速である (4K バイトあたり 6 ~ 20 ミリ秒)。

(2) ネットワーク接続時間の改善

TCP-stream は他のホストとのチャンネルの確立の手間が重い。例として、rsh のプロトコルを利用してサーバと交信する英和辞書アクセスクライアントを示そう。これは、エディタバッファから指示された英単語の語義・用例を、サーバからもらって表示するものである。1 回あたりのネット上のデータの転送量や、サーバでの仕事量はわずかであるが、当初これらよりもネットワークのチャンネルの確立に時間を要した。

これを軽減するため、サーバと通信する独立のプロセスを作り、一度確立したチャンネルを保持することとした。ただし、10 分間辞書を引かないと、チャンネルを解消する。これにより、初めて辞書を引くときには約 7 秒の時間がかかるが、二度めからはこれが約 3 秒に短縮された。これは、同じ辞書をローカルに引く時間とほぼ同じであり、ネットワークを通したオーバーヘッドは無視できることを表わしている。

以下に英和辞書アクセスクライアントの構成を示す。箱はプロセスを表わす。プロセス間はパイプストリームとメールボックスで通信している。



英和辞書アクセスクライアントの構成

3. ツール構築の手間の評価

UNIX のネットワークニュースシステムは、通信ソフトウェア、ニュースの記事管理、ニュースリーダなど C 言語で合計 4~5 万行のプログラムで構築されている。TAO/ELIS のニュースリーダ Znews (ニュースの配送や記事を読むためのプロトコルである NNTP のクライアント) は、TAO で約 3000 行である。これより、乱暴な評価ではあるが、数万行のシステムを 3000 行で利用したといえることができる。開発工数は 1 人月である。また、TAO/ELIS の IMAP (1 つのホストで一元管理されたメールボックスをアクセスするためのプロトコル) クライアントである Zmm は豊富な機能をもったメイラであるが、これも TAO で約 1600 行、半人月で開発された。

これらが少ない手間で開発できた理由として、クライアントとインタフェースだけを開発して、すでにあるソフトウェアを活用したこと、オブジェクト指向の継承機能を利用して、エディタの基本機能をそのまま使ってインタフェースを構成できたこと、効率のよいクライアントを構成するのに、各種のタイムアウトやプロセス間通信といった、TAO の並行プログラミングの機能が利用できたこと、などがあげられる。

4. おわりに

ここではソフトウェア資産を積極的に活用するためのクライアントの実現について述べた。

GNU Emacs エディタにおいても、画面エディタのインタフェースを利用した同様のメールリーダやニュースリーダが実現されている。だが、これはユーザインタフェースの向上や機能の充実が主目的であって、分散環境におけるツールの統合を主目的としているわけではない。

今回は LISP マシンでの実現を行なったが、ネットワーク機能が整備されたパーソナルコンピュータでも、同様のアプローチでソフトウェア資産の活用ができる。

最後に、開発の当初から使って頂き、多くの助言を下された利用者の方々に感謝いたします。

[参考文献]

- B.Kantor, P.Lapsley: Network News Transfer Protocol. A Proposed Standard for the Stream-Based Transmission of News, Request for Comments: 977 (1986)
- M.Crispin: Interactive Mail Access Protocol - Version 2, Request for Comments: 1064 (1988)
- 村上健一郎: オブジェクト指向による TCP/IP プロトコルの実現, コンピュータソフトウェア, Vol.6, No.1 (1989), pp.30-40.
- 天海良治: オブジェクト指向による画面エディタの部品化, 情報処理学会, 第 28 回プログラミングシンポジウム (1987)