遠隔資源への透過的アクセスを提供するコマンドインタプリタの実装

5F－6

本田 治[†]　多田 知正[†]　樋口 昌宏[‡]
[†] 大阪大学
大学院基礎工学研究科 情報数理系専攻
[‡] 近畿大学
理工学部 電気工学科

1 はじめに

現在，UNIX による分散計算機環境では，NFS などの分散ファイルシステムによってファイルに対する透過的なアクセスがある程度実現されている．しかしながら，遠隔計算機上のコマンドに関しては，透過的なアクセスが実現されていない．

そこで，本研究では，UNIX 上に，遠隔計算機上のコマンドを透過的に実行できるコマンドインタプリタの実装を行った．実装したコマンドインタプリタでは，ユーザはコマンド名を書くだけで，遠隔計算機上のコマンドをその位置を意識することなく利用することができる．

2 分散計算機環境と問題点

2.1 分散計算機環境

現在，複数の計算機がネットワークを介して接続されている分散計算機環境が一般的であり，ユーザーは複数の計算機上に存在する様々な資源を利用することで仕事を行う．その際，ユーザは遠隔計算機上の資源をその物理的な位置を意識することなく利用できることが望ましい．この性質を分散計算機環境の位置透過性 (location transparency)[3] という．以降では資源に対するそのようなアクセスを透過的なアクセスと呼ぶ．本研究では，分散資源への透過的なアクセスが可能な分散計算機環境の実現を目的とする．

2.2 UNIX による分散計算機環境

NFS は，現在 UNIX で広く一般的に用いられているファイル共有機構である．NFS で共有されたファイルは，透過的なアクセスが可能である．すなわち，UNIX ではファイルに関してはすでに透過的なアクセスが提供されていると言える．

ここで，UNIX 上で実行されるコマンドについて考える．ユーザはコマンドを実行することで，何らかのサービスを得る．すなわち，コマンドは一種の資源である．よって，資源としてのコマンドをコマンド資源と呼び，コマンドを実行する際にユーザが記述する文字列をコマンド名と呼ぶ．UNIX 上では，コマンド名はある実行ファイルの名前と等しい．よって，コマンド資源は一見，実行ファイルそのものであると思われる．もしコマンド資源がファイルであるなら，NFS で共有できることになる．しかしながら，実行ファイルは一般に，計算機のアーキテクチャ，接続されている物理デバイス等の様々な要因に依存しており，どの計算機でも実行できるわけではない．よって，コマンド資源を NFS で共有することは一般にはできない．すなわち，コマンド資源と実行ファイルは異なるものであると考えられる．分散計算機環境において，コマンド資源を特定するには，実行ファイルとそれが実行される計算機の両方が必要であり，実行ファイルのファイル名でコマンド資源を指定することはできない．このため，遠隔計算機上のコマンド資源 (遠隔コマンド資源と呼ぶ) を利用するためには，ユーザがコマンドを実行する計算機を陽に指定する必要があり，透過的であるとは言えない．

3 実装したシステム

本研究では，コマンド資源に対して透過的なアクセスを可能とするシステムを実装した．作成したシステムは，シェルと同様のコマンドインタプリタであり，ユーザはコマンド名を書くだけで，遠隔計算機上のコマンド資源をその位置を意識することなく利用することができる．コマンドインタプリタは，入力されたコマンド名を元に，アクセス可能なコマンド資源を見つけだし，実際にアクセスを行う．

3.1 コマンドテーブル

一般に，UNIX 上では，コマンドはローカルな計算機上で実行されるため，コマンド名から実行ファイルを直ちに特定できる．しかし，2.2で述べたように，分散計算機環境におけるコマンド資源の共有を考えた場合は，コマンド資源は，コマンド名から直ちに特定されない．よって，コマンド名からコマンド資源を得るためには，コマンド名とコマンド資源 (実行ファイルとそれを実行する計算機) の対応を管理する必要がある．そこで，表1に示すようなコマンドテーブルを用いて，これらの情報を管理する．

表 1: コマンドテーブル

コマンド名	実行ファイル	計算機
voice_comp	/usr/local/bin/vdc	eiger
	/usr/bin/vdcom	k2
gif_to_bmp	/usr/local/bin/g2b	eiger

実装したシステムは，このテーブルを検索することによって，コマンド名からコマンド資源を得る．あるコマンド名に対し，複数のコマンド資源が対応する場合，システムはその中の一つを選択する．一般的な UNIX と異なり，コマンド名は実行ファイルのファイル名と別に管理されているために，ある実行ファイルをコマンド資源として複数の計算機で共有するためには，あらかじめコマンドテーブルに登録しておくことが必要である．

3.2 遠隔コマンド資源へのアクセス

遠隔計算機上のコマンド資源を用いるには，なんらかの形で遠隔計算機上でコマンドを実行する必要がある．実装したシステムでは，各計算機上にコマンドの実行を代行するためのサーバを用意し，遠隔計算機 H のコマンド資源にアクセスする場合は H にコマンド資源に関する情報を送って，コマンドの実行を依頼するという手法をとっている．この方法では，遠隔計算機上で一つのコマンドを実行するたびに計算機間の通信が発生し，そのための遅延が生じる．これはシェルにおける対話的な利用においては大きな問題とはならないが，同一コマンドを繰り返すスクリプトを実行する場合などには，無視できない問題となる．

そこで，スクリプト全体が実行時に計算機間を移動しながら処理を行うという方針が考えられる．コマンド資源 C にアクセスするため計算機 H に移動したスクリプトは，C へのアクセスが終了した後も H にとどまり，以降の処理を継続できる．これにより，上記の問題をある程度解決できる．

実装したシステムでは，スクリプトを実行する際，スクリプトが移動しながらコマンド資源へアクセスを行う．スクリプトの移動のためには，移動前のスクリプトの実行状態を目的の計算機に移動し，移動先で実行状態を復元して，実行を再開する仕組みが必要である．作成したシステムでは，近年盛んに研究されている移動エージェント [1, 2, 5, 6] の考え方に基づき，この仕組みを実装している．

3.3 コマンド資源の選択

実装したシステムは，ユーザによって指定されたコマンド名から，適切なコマンド資源を見つけだし，実行する．しかしながら，1つのコマンド名に対し，複数のコマンド資源が対応する場合がある．この場合，システムは，どのコマンド資源に対してアクセスするのかを選択しなければならない．各コマンドは得られる結果は同じであっても，コマンド資源の存在する計算機の位置および処理能力，実行中に参照されるファイルの存在する計算機の位置など，さまざまな要因によって，コマンドの実行速度や，ネットワークへの負荷などが異なる．よって，システムが，複数の候補の中から適切なコマンド資源を選択することにより，実行速度の高速化や，ネットワーク負荷の抑制を達成することができる．また，スクリプトの実行時には，個々のコマンド資源の実行効率よりも，スクリプト全体の移動を少なくするようなコマンド資源の選択がより有効となる場合もあると考えられる．

現在の実装では，システムは最初に見つかったコマンド資源を選択する．コマンド資源のより適切な選択手法の検討は，今後の課題である．

4 まとめ

本稿では，従来の分散計算機環境上での遠隔コマンドの利用における問題点を指摘し，それを解決するため，分散計算機環境上でコマンド資源に対し，透過的にアクセスすることのできるシステムを実装した．

今後は，コマンド資源の選択方法の検討と実験による評価，他の分散ファイルシステム (例えば Coda[4]) に対応した実装等を行う予定である．

参考文献

[1] R. S. Gray : "Agent Tcl: A transportable agent system. In Proceedings of the CIKM Workshop on Intelligent Information Agents", Fourth International Conference on Information and Knowledge Management, 1995

[2] D. Johanson, R. V. Renesse, and F. B. Schneider : "An Introduction to the TACOMA Distributed System Version1.0", Technical Report 95-23, Depth. of Computer Science, Univ. of Tromosø and Corne ll Univ., Tromosø, Norway, June 1995

[3] K. Pradeep and Sinha : "Distributed Operating Systems Concepts and Design", IEEE PRESS,ISBN 0-7803-1168-X

[4] M. Satyanarayanan et al., Coda : "A Highly Available File System for a Distributed Workstation Environment", IEEE Trans. on Comp., vol. 39, no. 4, Apr. 1990.

[5] H. Peine : "An Introduction to Mobile Agent Programing and the Ara System", ZRI-Report, january 1997

[6] M. Straßer, J. Baumann, and F. Hohl : "Mole - A Java Based Mobile Agent System", Object-Oriented Programming ECOOP'96, M. Mühlaüser, ed., pp. 327-334, july 1996