7E-3

遅延実行によるマイクロ・メインフレーム結合の拡張とその適用

中島 周　　村上 和隆　　平賀 瑠美　　黒沢 隆
(日本アイ・ビー・エム株式会社 東京基礎研究所)

1．はじめに

パーソナル・コンピュータ（ＰＣ）とホスト・コンピュータを結合して連携処理を行うマイクロ・メインフレーム結合（ＭＭＬ）では、通信コネクションが常に設定されていることを前提にしている。我々は、ＭＭＬを利用した処理をコネクションが設定されるまで遅らせ、ＰＣ上では　バックグラウンドで要求を実行することによってより柔軟な結合を実現することを進めている。本稿ではこの遅延実行の機構とその適用例について述べる。

2．結合の拡張

ＭＭＬには、ＭＭＬだけの閉じた環境でサービスを提供するシステムと、ＰＣの使用環境をそのまま保存して任意のＰＣアプリケーションから使用できるサービスを提供するシステムとがある。どちらのシステムでも、ホストとＰＣ間にはコネクションが存在する状態でサービスを提供する。しかし、ホストとのコネクションはホストの稼働時間、通信路の障害などにより、ＰＣを使用していても存在するとは限らない。また、ＭＭＬ処理によっては、要求を依頼した時点で即時に処理する必要のないものもある。これらの処理要求をホストとのコネクションの有無にかかわらず発行し、コネクションが設立されたときに実行できれば、ＭＭＬ処理の可用性を向上させることができる。特に、仮想ディスク、仮想ファイル、仮想プリンタなどによってホスト資源をＰＣの資源の延長として見せるタイプのＭＭＬでは、ＰＣとホストとのコネクションをユーザに対して明示的には見せないので、このコネクションの仮想化はより有効である。

我々は、ＰＣからの要求をログに溜め、ホストとのコネクションを監視してコネクション存在時にＰＣからの要求をバックグラウンドでホストに送って実行させることにより遅延実行を実現する。ＰＣでのバックグラウンドの処理はＰＣの空き時間に処理を行うので、ユーザにはほとんど意識されない。

我々が現在ベースにしているＭＭＬでは、ＰＣからホストへの要求は送ることができるが、逆方向のホストからＰＣへは要求を送ることができない。そのため、あるイベントがホストで起きたときに特定の処理をＰＣに依頼するようなことはできない。そこで、これを実現するためにＰＣではバックグラウンドで一定時間ごとにホストをポーリングしてホストのイベントを監視する。もし検査したイベントの生起が確認されたならばそれをホストからの要求とみなしてＰＣが処理を行う。このようにして疑似的にホストからの要求も実現する。

3．遅延実行の機構

我々が実装した遅延実行の機構を図１に示す。遅延実行は、ＰＣ上の

- ●タスク制御部
- ●フォアグラウンド・アプリケーション
- ●バックグラウンド・アプリケーション

とホスト上の

- ●イベント検査ルーチン

によって構成される。

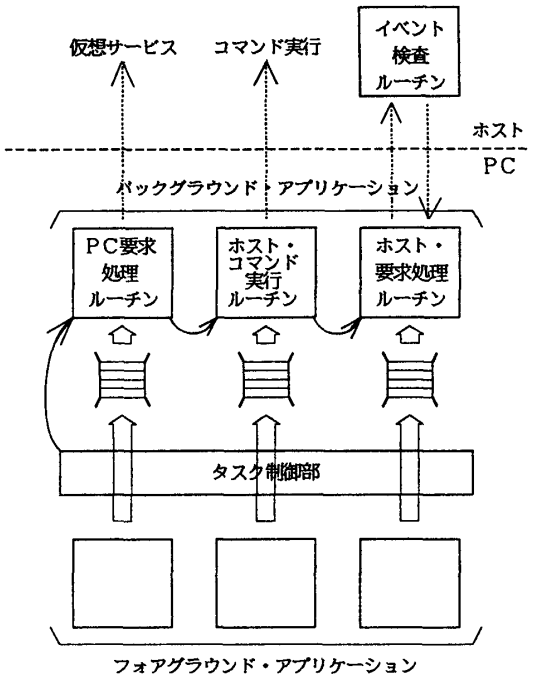

図１　ＭＭＬ処理の遅延実行の機構

An extended MML with a delayed execution mechanism, and its applications
Amane Nakajima, Kazutaka Murakami, Rumi Hiraga and Takashi Kurosawa
IBM Research, Tokyo Research Laboratory

ＰＣ側のＯＳがシングル・タスク（日本語ＤＯＳ）なので、バックグラウンド・アプリケーションが実行できるような環境を設定し、バックグラウンド・アプリケーションを呼び出すタスク制御部を導入した。このタスク制御部はタイマ割り込みルーチンとして実装される。タスク制御部の処理は以下のとおりである。

１．機能呼び出しやＢＩＯＳ呼び出しの排他制御
２．キー入力の機能呼び出しの書き替え
３．動作可能条件の検査
　ａ．コネクションの状態の監視
　ｂ．フォアグラウンド・タスクの状態の監視
　ｃ．バックグラウンド・タスクの状態の監視
　ｄ．最後のキー入力からの経過時間の検査
　ｅ．最後の機能呼び出しやＢＩＯＳ呼び出しからの経過時間の検査

２はキー入力待ち時のバックグラウンド・タスク実行のために必要となる。３－ａ、３－ｂ、３－ｃにより、バックグラウンド・タスク実行のための最低の必要条件が検査される。３－ｄと３－ｅは、バックグラウンド・タスクをフォアグラウンド・タスクの空き時間に実行するために必要となる。つまり、キー入力や機能呼び出しが一定時間ないときは、フォアグラウンド・タスクの処理が休止中だと判断してバックグラウンド・タスクを実行する。

フォアグラウンド・アプリケーションは、目的とするＭＭＬ処理のために、処理要求をログの形で作成する。このログはバックグラウンド・タスクが遅延実行する。

バックグラウンド・アプリケーションは、タスク制御部によって起動され、ＰＣからホストに対する処理をログに基づいて実行する。複数のバックグラウンド・アプリケーションをチェインすることができる。一度のバックグラウンド・タスクの実行ではこの中の１つが実行される。現在、バックグラウンド・アプリケーションにはＰＣ要求処理ルーチン、ホスト・コマンド実行ルーチン、ホスト要求処理ルーチンの３種類がある。

ＰＣ要求処理ルーチンは、ＰＣからホストに対する要求のうち、ＭＭＬの仮想サービスを利用するものを処理する。

ホスト・コマンド実行ルーチンは、ログから読んだホスト・コマンドをホスト・コマンド実行のキャラクタ・デバイスに書き込む。このデバイスはホストとの通信に開放されているＰＣ上のデバイスで、ここに書き込んだ文字列はホストに送られて実行される。逆にホストの出力はこのデバイスを読むことによってＰＣに得られる。

ホスト要求処理ルーチンはホストからＰＣに対する処理要求を実行する。我々が現在ベースにしているＭＭＬでは、ＰＣからホストへの要求は送ることができるが、逆方向のホストからＰＣへは要求を送ることができない。そこで、ホスト要求処理ルーチンではバックグラウンドで一定時間ごとにホストのイベント検査ルーチンを起動する。ホスト要求処理ルーチンはイベント検査ルーチンの応答を読み、それに対応した処理を行う。このように、ホストの状態のポーリングにより疑似的にホストからの要求を実現する。

４．適用例

現在までにファイルの自動複製機能[１][２]と、ＰＣファイルのホスト上のユーザへの／からの送信／受信機能を作成した。

ファイルの自動複製では、ＰＣ上の指定したディレクトリ下のすべてのファイルをホストの仮想ディスク上に遅延実行で自動的に複写する。

ＰＣでのファイル送信要求はフォアグラウンド・アプリケーションとして作られている。送信要求時に作られたログに基づきＰＣ要求処理ルーチンがホストへのファイルの複写を行い、ホスト・コマンド実行ルーチンがホストでの送信を実行する。ＰＣでのファイル受信操作は、ホスト要求処理ルーチンとイベント検査ルーチンによって行われる。ＰＣ上のホスト要求処理ルーチンが一定時間間隔でホスト上のイベント検査ルーチンを呼び出して、ホストにファイルが送られているか調べる。もし送られていれば、ホスト要求処理ルーチンがホスト・コマンドを実行してまずホストで受信する。その後、ファイルをＰＣに複写する。

これらの処理は、ホストとのコネクションがないときは要求だけをログに保存し、後でバックグラウンドで実行される。よってユーザはコネクションの有無を意識しない。

５．おわりに

現在はバックグラウンド・アプリケーションの３つのルーチンの間には固定的な優先順位がついている。また、各処理で使用するログは分離されている。そのため、この機構を汎用に使用すると、ログの実行順序が発行順に処理されないことが起きうる。タイム・スタンプを使用することも考えられるが、ホストとＰＣの時刻の差や、ＰＣの時刻が容易に変更されることを考えるとこれは適当ではない。そこで、すべてのログを１つにまとめることにより、これを解決し、さらに多くの処理にこの機構を適用する予定である。

参考文献

[１]村上他：ＭＭＬ環境における自動ファイル複製システム(1) 概要、情報処理学会第37回全国大会
[２]平賀他：ＭＭＬ環境における自動ファイル複製システム(2) 技術的側面、情報処理学会第37回全国大会