LOTOS仕様の実装法に関する一考察

4T-5

程子学　高橋薫　白鳥則郎　野口正一
(東北大学電気通信研究所)

1. はじめに

システムの開発においては厳密でかつ正確な記述を可能とする形式記述技法(FDT)による仕様化が望ましい。この背景の下で、通信システムの仕様化を目的としたFDTであるLOTOSがISOによって提案されており、今後広く通信プロトコルの開発に用いられることが期待されている。したがって、LOTOSによる仕様を計算機システム上で実行できるプログラムに変換し、インプリメントの支援を行うことは通信ソフトウエアの生産性の向上に大きく寄与する。本稿では、C言語とUNIX環境の機能を用いたLOTOS仕様の実装法について考察する。

2. LOTOS仕様の実装法

2.1　概要

半自動的な実装方針を取る。すなわち図1のように二段階に分けて実装する。まず、人手により、仕様の動作部に対して実装情報の付加、プロセスの合併などの処理をし、動作部'に変換する。一方、データ部(ADT)に対して意味解析をし、その解析結果に基づいて(意味を保存しながら)C言語の型を使って同じ機能を実現できるようにコーディングする。第一段階の過程は今後AI的な手法を用いて対話的に行うことを計画している。

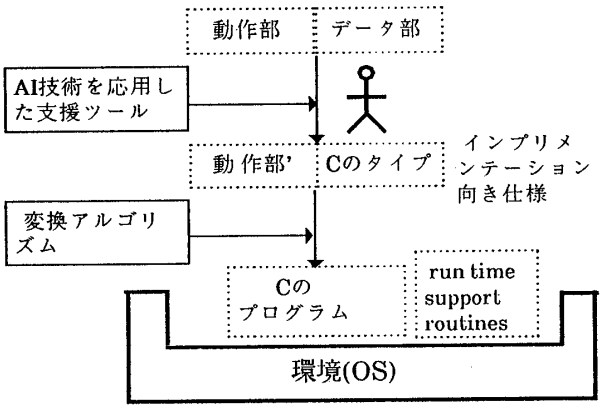

図1　本実装法の全体概要

第二段階では、前の段階で得られた仕様(インプリメンテーション向き仕様)を変換アルゴリズムを用いて、自動的にシステム環境の上で実行可能なプログラムに変換する。

2.2　動作部の実装

2.2.1 基本方針

並列的に動作するLOTOSのプロセスをUNIX上のユーザプロセスと対応付け、LOTOSのランデブー通信をOSのプロセス間の通信用ツールであるソケットを使って実現する。プロセスにおけるイベントの同期の達成のため、イベントの記述は次の図2の手続に変換され、実装される。

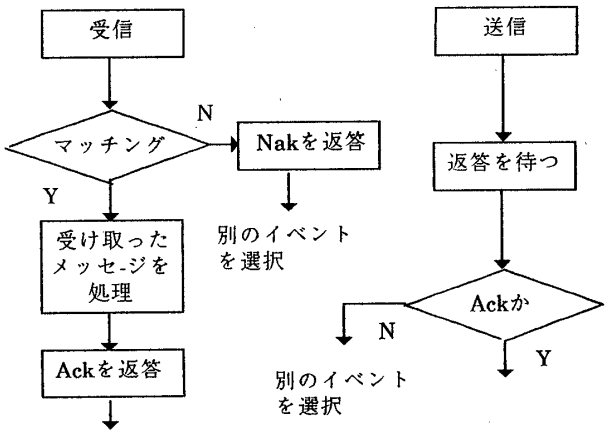

図2　イベント同期メカニズム

ここで送信、受信はUNIX上のwrite()、read()システムコールを用いる。受け手の方で値のマッチングテスト、メッセージのパッシングが行われる。マッチングテストが成功すれば、Ackを返答

Implementation Method for LOTOS Specifications
Zixue CHENG, Kaoru TAKAHASHI, Norio SHIRATORI, Shoichi NOGUCHI
Research Institute of Electrical Communication, Tohoku University

し、次のイベントの処理にすすむが、失敗の場合には、Nakを返答し、別のイベントを選択する。送り手の後処理で返答の内容によって次のイベントの処理に進むか別のイベントを選択するかが決まる。

2.2.2 LOTOSオペレータの実現

プロセス間の関係とプロセス内部の動作を表すオペレータをCのステートメントあるいはUNIXのシステムコールを使って実現する。直接にCのステートメントあるいはUNIXのシステムコールで実現できない場合はランタイムサポートルーチンの支援を加えることによって実現する。

オペレータstopはOSのユーザプロセスを終結させるシステムコールexit()を用いて実現する。action - prefixはイベントの処理文の前後順序と対応させて実現する。choiceはif文を用いて実現する。独立動作を表す並列オペレータに対しては、fork()とexecl()システムコールを使って、二つ完全に独立なプロセスを生成し、OSの管理のもとに並列に動作させる。同期並列オペレータに対しては、fork()とexecl()システムコールを使って、二つ完全に独立なプロセスを生成してから、その間にソケットをはって、write()、read()システムコールを用いて、ランデブー通信を実現する。instantiationはシステムコールexecl()と対応させる。割り込みオペレータはkill()とsignal()システムコールと、順次合成オペレータはexecl()システムコールと対応させて実現する。

2.2.3 多重ランデブー通信の実現

複数のプロセスがランデブー通信する場合にはまず、fork()とexecl()というシステムコールを使って、複数のユーザプロセスを生成し、その間にソケットをはる。さらに値のマッチングとメッセージのパッシングを行うために、同期イベントを対応した手続に変換する。

例えば,P1|[g]|P2 |[g]| P3のような仕様に対して、図3のようにまずfork()とexecl()というシステムコールを使って、三つのユーザプロセスを生成してからP1とP2、P2とP3の間にそれぞれソケットをはる。そしてP3から同期ゲートgに関するデータをP2に送り、P2が受信してから、データのマッチングをとり、同期条件が満足されれば、P3から受け取ったデータをさらにP1に送る。P1は受信してから、P2と同様のデータ処理をし、同期条件が満されれば、P2に返答する。P2はその返答を受けてから、さらにP3に返答する。

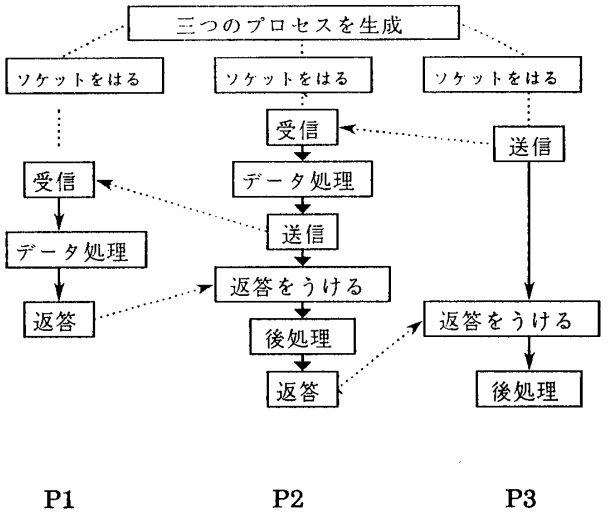

図3　多重ランデブー通信の例

2.3　データ部(ADT)について

ADTに対しては人間が型ごとに解釈し、意味上の各値についてシステムのあるメモリの状態と一対一に対応させる。そのメモリはC言語のある型の変数に割当てられるもので、ソートごとに一つのC言語の型として実現する。

3. 考察

OSの機能を使って、LOTOSプロセスの生成、消滅とランデブー通信を実現するという方法は仕様との対応が明確であり、変換もしやすいと考えられる。一方、仕様中のプロセスの数が多い場合、生成されたユーザプロセスの数も多くなり、実行のスピードが遅くなりうる。そのため、仕様はRefinementの手法などでプロセスを合併することが望ましい。

参考文献

(1) ISO, "Information processing systems － Open Systems Interconnection － LOTOS － A formal description technique based on the temporal ordering of observational behaviour," ISO 8807 (1989).

(2) 野村, 長谷川, 瀧塚, "LOTOS実行系の並列処理環境," 信学技報, COMP89-4 (1989).