

1V-10

拡張TSチャートによる リアルタイムシステムの解析技法について

吉田 聡 大原茂之
東海大学

1. はじめに

リアルタイムシステムを1つの記号体系で表現することは困難であり、目的に応じて複数のダイアグラムを使い分けて記述する¹⁾。このため、ダイアグラムによって記述や解析手法が異なり、リアルタイムシステムの各機能を統合して解析することが困難である。

本報告では、タスクの状態遷移、スケジューリング、タイムアウト、およびプライオリティインバージョンの解析が可能な拡張TSチャート^{2),3)}を用いた、リアルタイム同期、リアルタイム通信の解析手法について述べる。

2. オブジェクトの定義

時間制約のあるタスクは内部に時計をもち、サバイバル時間、残り処理時間などをカウントするオブジェクトとしてとらえる（以降、タスクオブジェクトと称す）。同様に、時間制約のある通信メッセージも内部に時計をもち、メッセージのサバイバル時間をカウントするオブジェクトとしてとらえる（以降、通信メッセージオブジェクトと称す）。

他にオブジェクトとしては、タスク生成、タイムアウト処理タスク生成、ソータ（1, 2）、I/O資源管理、CPU管理、メールボックス管理の各オブジェクトで構成されている。

各オブジェクトとそのメッセージの定義は次のとおりである。

【通信メッセージ】タスク間通信におけるメッセージであり、タスクオブジェクト、メールボックスオブジェクトの間を移動するオブジェクトである。

送信メッセージ：TimeoutMes

【タスク】タスク生成、タイムアウト処理タスク生成、ソータ1、ソータ2、CPU管理の各オブジェクトの間を移動する。したがって、タスクオブジェクトの移動そのものが各オブジェクトのメッセージと考えることができる。

送信メッセージ：I/O_req, Timeout, Mes_req

受信メッセージ：I/O_reply, I/O_Noreply

送受信メッセージ：CPU_req, CPU_release,
CPU_Noreply, RES_req, RES_reply,
RES_Noreply, RES_release

【タスク生成】イベントが発生するとそれに対応するタスクを生成する。

【タイムアウト処理タスク生成】タイムアウトのメッセージがあれば、それに対応するタイムアウト処理タスクを生成する。

受信メッセージ：Timeout

【ソータ1】タスクオブジェクトの実行可能状態を意味し、優先度でソーティングする。

【ソータ2】タスクオブジェクトの待ち状態を意味し、優先度でソーティングする。

【I/O資源管理】I/O資源の利用許可・禁止を行う。

送信メッセージ：I/O_reply, I/O_Noreply

受信メッセージ：I/O_req

【CPU管理】このオブジェクトに渡されたタスクが、CPUを利用することができる。

【メールボックス管理】タスクの通信メッセージを管理する。

受信メッセージ：Mes_req

3. リアルタイムモニタの記述

図1に拡張TSチャートによるオブジェクト表現の例、図2に拡張TSチャートで記述したリアルタイムシステムを示す。この図を用いることで、タスクの状態遷移、スケジューリング、タイムアウト、プライオリティインバージョンなどの解析が可能となる。

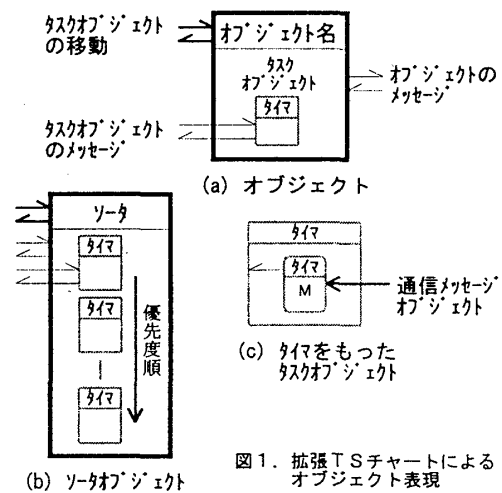


図1. 拡張TSチャートによるオブジェクト表現

4. システムの解析

4.1 資源獲得

横取り可能な資源R₁を利用するタスクT₁は、すべて

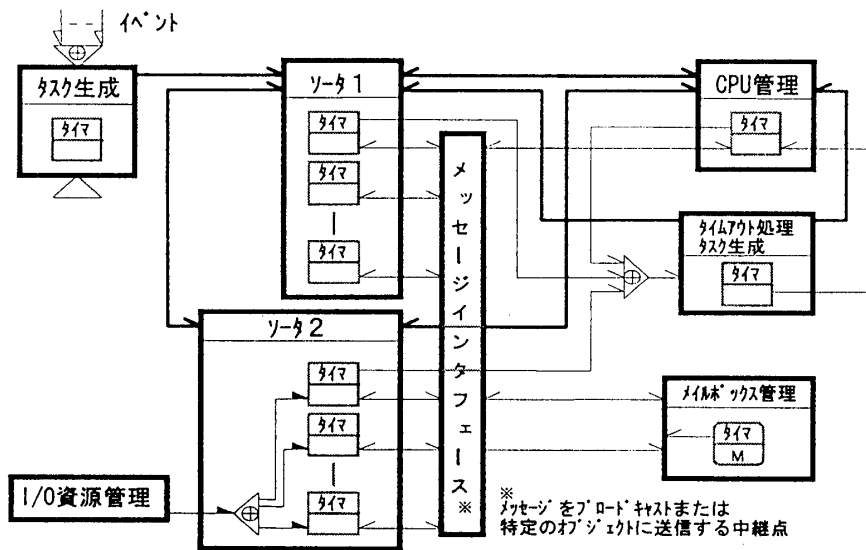


図2. 拡張TSチャートで記述したリアルタイムシステム

のタスクにメッセージ RES_req (T₁の優先度, 現時刻, T₁, R₁) を送信する。

タスク T₁から RES_req メッセージを受信したタスク T₂は, 資源 R₁を使用していなければ T₁にメッセージ RES_reply (現時刻, T₂, R₁) を送信する. 資源 R₁を使用していて,

T₁の優先度 < T₂の優先度ならば, T₂は T₁に RES_NOREPLY (現時刻, T₂, R₁) を送信する.

T₁の優先度 > T₂の優先度ならば, T₂は R₁を解放し, T₁に RES_reply (現時刻, T₂, R₁) を送信する.

T₁がメッセージを送信したときの時刻よりも新しい時刻のパラメータをもつメッセージ RES_reply をすべてのタスクから受信したとき, T₁は R₁を利用できる.

4.2 リアルタイム同期

排他資源 R₂を利用するタスク T₃は, すべてのタスクにメッセージ RES_req (T₃の優先度, 現時刻, T₃, R₂) を送信する.

タスク T₃から RES_req メッセージを受信したタスク T₄は, T₄がもつ待ち行列 QUEUE (T₄) にメッセージ RES_req を優先度順に挿入し, 資源 R₂を使用していなければ T₃にメッセージ RES_reply (現時刻, T₄, R₂) を送信し, 資源 R₂を使用していれば T₃にメッセージ RES_NOREPLY (現時刻, T₄, R₂) を送信する.

T₃の RES_req が QUEUE (T₃) の先頭にあつて, T₃が送信したメッセージのパラメータ “現時刻” よりも新しい時刻のパラメータをもつ RES_reply をすべてのタスクから受信したときに限り, T₃は資源 R₂を利用することができる.

資源 R₂を解放するときには, T₃は自分の RES_req メッセージを QUEUE (T₃) から除去し, 他のタスクに RES_release (T₃, R₂) を送信する.

T₄が T₃のメッセージ RES_release を受信したときには, T₄のメッセージ RES_req を QUEUE (T₄) から除去する.

これによって, Lamportのアルゴリズムと同様に相互排他を実現することができる.

また, 図2を用いることでデッドロックの解析を行うことも可能である.

4.3 リアルタイム通信

タスクはメッセージを送信するときに, 通信メッセージオブジェクトを生成する.

タスク T₇はメールボックスにある通信メッセージ M₁を受け取るとき, Mes_req (T₇, M₁) をメールボックス管理オブジェクトに送信する.

メールボックス管理オブジェクトは M₁を T₇に移動させる.

M₁がサバイバル時間に達したときは, TimeoutMes (M₁) メッセージを各オブジェクトに送信する.

TimeoutMes メッセージを受信した各オブジェクトは, M₁を消滅させる.

5. おわりに

拡張TSチャートでリアルタイムシステムを記述することにより, スケジューリング, タイムアウトなどの解析と統合して, リアルタイム同期, リアルタイム通信の解析ができることを示した. 今後は, これらの解析を自動化する手法を検討し, リアルタイムシステムの設計支援システムの実現をめざす.

謝辞 本研究を進めるにあたり, 日頃お世話になっている本学電子工学専攻主任飯田昌盛教授に感謝の意を表します.

参考文献

- 1) 伊藤, 杵島: リアルタイムシステムにおけるネット指向開発技術の適用, 情報処理, Vol. 34, No. 6, pp747-760 (Jun. 1993).
- 2) 吉田, 大原: 拡張TSチャートによるタスクスケジューリングに関する一提案, 情処学第48回全大2H-6 (1994).
- 3) 吉田, 大原: 拡張TSチャートによるリアルタイムシステムの記述手法について, 信学'94秋大D-52 (1994).