1W−4

産業用データベース・ミドルウェアの開発
— 実時間トランザクション管理機構 tact —

島川 博光† 西村 健† 竹垣 盛一†

†三菱電機(株)産業システム研究所

1 まえがき

実時間トランザクション管理は実時間データベースにおける核部分である．しかし，その実時間スケジューリング・ポリシは適用問題ごとの物理的制約や仕様による制約により変更できることが望ましい．我々はCPU使用効率は悪いが静的なスケジューリング技法であるため最も実用的とされるRate Monotonic Analysis(RMA)[1]とCPU使用効率を良くする動的なスケジューリング技法であるFrame Scheduling[2]のいずれのスケジューリング・ポリシでも選べるような実時間トランザクション管理機構tact(Tactics to Activate and Control Transactions)を開発している．本稿ではtactの特徴と実現法について述べる．

2 tactの特性

tactには以下のような特性がある．

- 協調型システム構成をとりながら，部分デバッグを可能としている．
- スケジューラとトランザクション起動機構の独立性が高い．
- 各実行主体が他数のモード間を遷移できる．

2.1 協調型システム

tactでは，トランザクションを実行する複数エージェントが協調動作することによってデータベース・サーバとしての機能が実現される．例えばRTDS実現のためには，プラントからのデータ獲得のエージェントやEWS, PCへのデータ提供のためのエージェントが存在する．

tactは，サーバと，エージェントが使用するスタブ関数の集合として実現されている．tactの利用者は各エージェントが実行すべき手続きを記述する．tact利用者にはサーバは隠蔽されている．代わって，tact利用者には実時間トランザクションの登録など，サーバの機能を利用するためのスタブ関数が提供される．サーバを立ち上げた時点でこれらスタブ関数が利用可能となる．この意味でtactは利用者にとって環境として位置付けられる．

tactのサーバ内には，スケジューラやトランザクション起動機構，スケジュールを保持するリング型共有メモリが存在する．tactのサーバからは各エージェントに対してスケジュールされたトランザクション実行のための起動メッセージが送られる．このメッセージを受けとったエージェントがトランザクションを実行する．

このような協調型のシステム構成をとる場合，すべてのモジュールを開発してからでないとデバッグができないようでは実用的なソフトウェアの開発はできない．そこで，tactの実装では，エージェントがメッセージ送受信のために使用するメッセージ・キューや共有メモリなどの資源をtact環境の立ち上げ時に割り付ける．

2.2 独立性

tactでは，スケジューラとトランザクション起動機構は共有メモリを介して非同期にアクセスできる．スケジューラは作成したスケジュールをリング型の共有メモリに書く．そのスケジュールにしたがって，トランザクション起動機構がトランザクションを起動する．リング型の共有メモリでスケジューラが書き込みを行なっている部分とトランザクション起動機構が読み出しを行なっている部分に重なりはなく，これらは独立に動作できる．

共有メモリ上のスケジュールに対してスケジューラは生産者，トランザクション起動機構は消費者の関係にある．独立性が高いためにRMA[1], Frame Scheduling[2]など複数のスケジューリング・ポリシから選択されたポリシに基づくスケジューラをtactに適応するできる．アプリケーションからはこれらのポリシやスケジューラは隠蔽されており，ポリシやスケジューラを変更してもアプリケーションには影響はない．よって問題に適したポリシが選択できる．

2.3 モード遷移

フォールト・トレラントなシステム構築のために制御システムは通常二重化される．すなわち，制御コマンドを出す主系に，常に切替え可能な状態の従系が並走している．通常，これらの切替えはそれぞれの系のモードを切替えることにより実現される．制御システムにデータベースを適用する場合には他の構成要素と同様に二重化機能をデータベースが備える必要がある．二重化のためにtactはモード遷移機能を支援する．

tactエージェントが実行すべき手続きは，モードとメソッドの組として指定される．tactのサーバはモード遷移のためのメッセージをtactエージェントに送ることができる．tactエージェントはモード遷移メッセージを受けとれば自らのモードを変える．さらに，tactエージェントはトランザクション起動メッセージを受けとれば自らのモードを調べ，そのモードに対応したメソッドを実行する．

Development of Middleware for Industrial Database Systems – Real-Time Transaction Monitor: tact –

Hiromitsu Shimakawa, Takeshi Nishimura, Morikazu Takegaki

Mitsubishi Electric Corp.

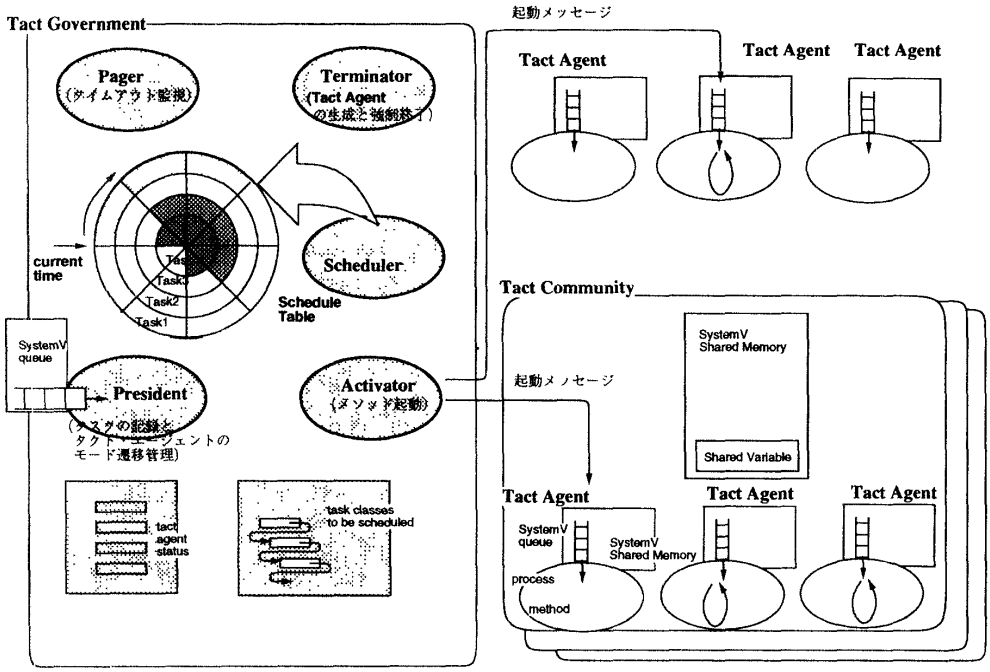

図 1: tact 全体構成

3 構成

tact 環境は tact agent とサーバである tact government からできている．tact agent は共有メモリを使って互いに情報を共有する tact community を形成することができる．

3.1 tact agent

UNIX プロセスに，キュー，共有メモリ，セマフォを付加し，これをトランザクション実行主体とする．tact ではこの機能付加されたプロセスを tact agent と呼び，トランザクション実行の単位と考える．

tact agent はオブジェクト指向型の応答型タスク実行を行なう．tact agent 基本モデルはキューによるメッセージの受信とそれへの応答である．したがって，周期的にタスクを実行している tact agent は，内部でタイマ処理をしているのではなく，周期的に外部からキューにメッセージを受信することによって，あたかも周期的にタスクを実行しているように見えるだけである．このようなモデルを採用したのは

- モード遷移とタスク実行を無矛盾な形で統合するためと，
- 外部からのメッセージを受けとった時のコールバック関数だけを tact ユーザに意識させるため

である．

3.2 tact government

tact agent はサーバ tact government の支配下におかれる．tact government は以下のような tact government 内の共有メモリ上のデータとプロセスから構成される．

スケジュール・テーブル タスクのスケジュールを表す．

tact agent 状態テーブル tact agent の状態を示す．

タスク・クラス・リスト スケジュールされるべきタスクを示す．

スケジューラ タスクをスケジュールするプロセス

activator tact agent にタスク実行の開始を指示するプロセス

terminator 常駐の tact agent を生成するとともに，重要な tact agent が異常終了した時にシステム内のすべての tact agent を強制終了させ，システムの暴走を防ぐプロセス

president 新しいタスクの受け付け，タスクを実行する tact agent の状態管理やモード遷移管理を行なうプロセス

pager tact agent にタイムアウトを指示するプロセス

4 あとがき

本稿では実時間トランザクション管理機構 tact の特性と実装を説明した．tact は実時間 OS 上で開発されており，現在，試験を進めている．また tact をさまざまな OS 上で実装することも検討している．

参考文献

[1] C.L.Liu and J.W.Layland, Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, *J. ACM*, Vol.20, No.1, pp.46-61, 1973

[2] K.Ramamritham, et.al, Efficient Scheduling Algorithms for Real-Time Multiprocessor Systems, *IEEE Trans. Parallel and Distributed Systems*, Vol.1, No.2, 1990, pp.184-194