生産システムにおける分散フォールト・トレランスの実現†

1S－7

*中田剛司　垂水宏文　田辺繁美　福留五郎
オムロン(株) コントローラ研究所第1技術開発課

1　はじめに

コンピュータのタウンサイジングに伴い、従来の集中システムに代わり対コスト性や拡張性に優れた分散システムが主流になってきた。我々は工場の生産システムにこの分散システムを適用した分散型CIMを提案、推進してきた。

一方、近年、大規模・複雑化する生産システムでは、コンピュータ等の故障によるシステム停止は大きな機会損失につながる。そのため、故障に対してすみやかに復旧を行ないシステム停止時間を最小限にするようなフォールト・トレランス性が求めらている。

このような背景のもと、分散型CIMにおけるフォールト・トレランスの実現を試みた。その結果、分散システムの持つ対コスト性や拡張性を継承しつつフォールト・トレランス性を実現できた。本稿ではその概要を報告する。

2　システム概要

今回開発したシステムは、工程管理や生産計画等をジョブとするセル、エリアレベルのコンピュータをネットワーク上の他のコンピュータにより2重化するシステムである。図1に本システムの構成を示す。コンピュータCが

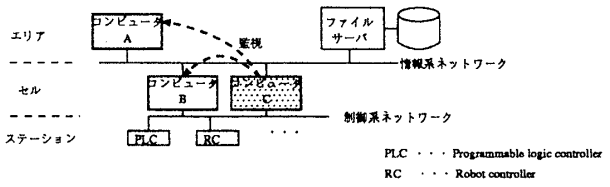

図1: システム構成

コンピュータA,Bを監視し、フォールトが発生した系のジョブをBが引き継ぐ。このとき、引き継ぎ後の現有系A,Bと待機系Cのデータの共有は両者がファイルサーバ上のデータをリモートアクセスすることで行なっている。

†Development of distributed fault tolelant system for manufacturing system.
Takeshi Nakata, Hirofumi Tarumi, Shigemi Tanabe, Goro Fukutome
OMRON Co. Intelligent Controller Lab

3　冗長方式

一般に冗長系の実現方式は常用冗長方式と待機冗長方式に大別される。

前者は複数の系が同一処理の処理結果を比較することでフォールトを検出し、その後は正常な系のみの処理を続行するものである。引き継ぎ時間が極めて小さくジョブの連続性が高い反面、フォールト系特定のために3つ以上の系が必要になりコスト的には高くつく。

後者は、現有系と待機系から構成され、通常動作時は待機系が現有系を監視する。待機系は現有系フォールトを検出した場合、現有系のジョブを引き継ぎ実行する。フォールト発生からの切り換え時間は常用冗長方式に劣るが、1台の待機系で複数の現有系を二重化できるなど対コスト性に優れる。

今回は、分散システムの対コスト性を重視するため、待機冗長方式を採用した。

4　フォールト検出

本システムでは2つの方法によりフォールト検出を行なった。

(1) アライブ通信。
現有系が定期的に待機系に対してアライブ信号を送信し、その受信が一定時間途切れることで待機系は現有系をフォールトと判定する。アライブ通信の実現においては次の2点について考慮した。

1-1) アライブ通信路。
アライブ通信を行なう通信路は、アライブ通信専用の通信路を設ける方法が一般的である。しかし、専用通信路はシステムの分散性の損失やコストアップにつながるため今回は図1における情報系ネットワーク上でアライブ通信を行なった。

1-2) 通信装置のフォールト。
ネットワークをアライブ通信路とした場合、待機系自身の通信装置がフォールトになった場合でも待機系は現有系フォールトと判定して引き継ぎ処理を行なってしまう。これを防ぐため、待機系ではアライブ通信が途切れた時に現有系以外の他ノードに対して折り返し通信テスト実

行し、その結果が正常である場合に引き継ぎ処理を行なうようにした。

(2) 現有系の自己診断。
情報系ネットワークを用いてのアライブ通信だけでは、制御系ネットワークの通信装置のフォールトを検知できない。そこで、制御系ネットワークの通信装置については現有系で自己診断を行ないフォールト発生時はそれを待機系へ通知するようにした。

5 フォールト系切り離し

待機系は現有系のジョブ引き継ぎの際に、ネットワークアドレスも同時に引き継ぐ。そのため、フォールトを起こした現有系はネットワークから切り離されていなければアドレスの重複が発生する。本システムでは、離脱コマンドのある通信装置にはコマンドの発行により、離脱コマンドのない装置に対しては他ノードで使用されていないダミーアドレスに設定し直す方法でネットワークからの切り離しを行なった。

6 ジョブ引き継ぎ

待機冗長方式では引き継ぎ時にジョブの内部状態(メモリ上の内部データ)と外部状態(ファイルデータ)との整合をとる必要がある。その手法としては通常時にジョブの内部状態をチェックポイント毎に現有系から待機系へ送信するチェックポイント方式が一般的である。しかし、この方式ではジョブの内部状態送信のためのオーバーヘッドや送信中のフォールトに対して不連続になる等の問題がある。

そこで、本システムでは引き継ぎ対象をトランザクション処理型の構造を持つアプリケーションとし、引き継ぎの際に外部状態をロールバックすることで内部状態との整合をとる方式をとった。

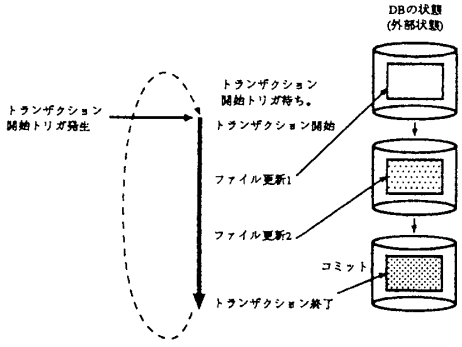

図 2: トランザクション処理型アプリケーション

トランザクション処理型アプリケーションは図2に示すように最初ステートレスな待ち状態にあり、開始トリガをうけて一連のファイル更新処理を行なう。この更新処理のファイルへの反映はトランザクション終了時のコミット時になされる。

このような構造のアプリケーションを対象とし、図3に示すような手順でジョブの引き継ぎを行なう。

(1) フォールト検出後待機系では、フォールトが発生した時点で処理途中であったトランザクションがないかを検査する。

(2) 処理途中であったトランザクションがあればそのアボート処理を行ない、ファイル状態(外部状態)をトランザクション開始時点までロールバックする。

(3) アプリケーションを再起動する。

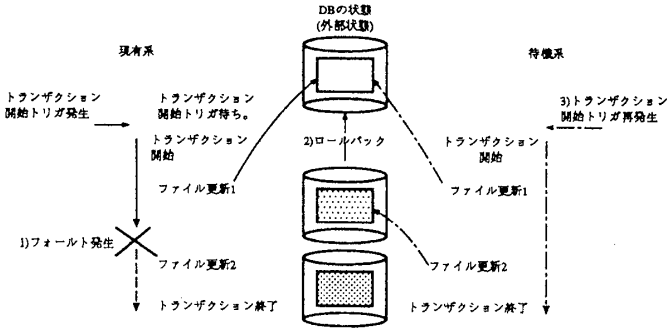

図 3: アプリケーションの引き継ぎ

7 おわりに

今回のシステムでは1台の待機系で複数台の現有系を二重化でき、また、ソフトウェアのアドオンにより二重化システムを構築することが出来る。これにより、分散型CIMの持つ対コスト性や拡張性を継承しつつシステムにフォールト・トレランス性を持たせることが可能となった。

今後の課題として以下の項目について検討を進めていきたい。

(1) 待機系が現有系を監視しつつ別のジョブを実行できること。

(2) 引き継ぎ時間の短縮。

参考文献

[1] 当麻 喜弘, "フォールト・トレラント・システム論", 電子情報通信学会 (1990)