システム進化支援環境 リセプティブプラットホーム

6E−1

桧垣博章 森保健治 奥山浩伸 平川豊 市川晴久
NTT ソフトウェア研究所

1 はじめに

近年の通信技術、計算機技術等の発達によって、交換網、OA網、コンピュータネットワークなどに代表される大規模ネットワーク上で、広域分散する多種多様のリソースを利用した高度で多様なサービスを実現する大規模分散システムの構築が可能になっている。このような大規模分散システムは、定常状態で運用されることは希であり、常に何らかの「変化」が生じていると考えられる(図1)。この変化はその起因となるものによって、大きく次の2種類に分類できる。

ユーザ主導型変化: システムに対するユーザの要求は、多種多様でありかつ運用中に経時変化する。これに対応するためのシステム構成要素(アプリケーションプログラム、リソース)の追加や変更による変化を「ユーザ主導型変化」と分類する。

システム主導型変化: 大規模分散システムを構成する非常に多数のリソースが、常に正常であるとは限らない。これらの故障はシステムあるいはそれを取りまく環境の作用(災害による電源断等)によるものであり、「システム主導型変化」と分類する。

いずれに対しても、変化が及ぼす影響によってサービス提供の停止、システム誤動作が発生しないことを保証しなければならない。この影響は次の2種類に分類できる。

直接的影響: あるサブシステムへの機能追加のために、ある一定時間のサービス提供停止を要することがある。このように、変化するサブシステムそのものへの影響を「直接的影響」と分類する。

間接的影響: ある機能追加したサブシステムと他の機能追加されていないサブシステムとの通信によって、システムが異常状態に陥る可能性がある。このように、直接的影響を受けないサブシステムへ通信によって波及していく影響を「間接的影響」と分類する。

本論文では、システムの様々な変化による影響を受容し、サービスを正しく安定的に提供する環境を実現するための分散プラットホーム、「リセプティブプラットホーム」を提案する。

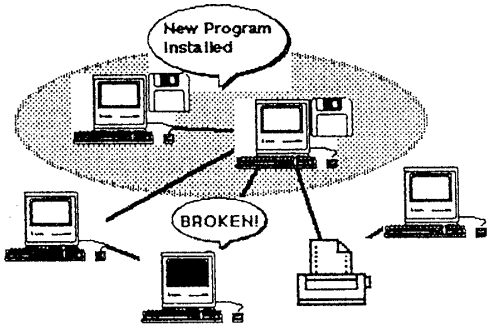

図1: 分散システムにおける変化

2 リセプティブプラットホームが解決すべき諸問題

リセプティブプラットホームは、前章で述べたシステムの変化に対応する機能を持つ必要がある。システム主導型変化への対応については、フォールトトレランス機能が該当し、我々はプロセス間通信に関してグループ通信を用いた実現技術[3]を提案している。そこで本論文では、ユーザ主導型変化による問題への対処技術について述べる。

ユーザ主導型変化とは、機能の新規導入と高度化、あるいは使いやすさの向上といった要求の実現結果である。その実現方法は、システムを構成する運用中の実行実体(旧版プロセス)を、機能追加や機能変更がなされた新しい実行実体(新版プロセス)に置き替えるという「改版」操作であると考えられる。例えば、固定されていたプリンタの出力先を複数から選択可能にする変更や、データベース上のデータ形式の変更に伴なう入力プログラムの変更などは、改版によって実現される。改版操作の実行には、次のような問題がある。

直接的影響による問題: 改版操作を複数のプロセスで同時に行なう場合を考える。改版に関わる全てのプロセスを停止する方法は、大規模分散システムにおいてはコスト的な問題が大きく、また、稼働中サービスの継続的提供が不可能になる。そこで、新版プロセスと旧版プロセスが混在する「過渡状態」の形成によって、これを解決する。過渡状態においても、システムが異常状態に陥ることを回避しなければならない。

間接的影響による問題: 大規模分散システムに対するユーザの要求は多種多様である。そこで、機能追加や機能変更がシステム全体に作用することを強要せず、サブシステムごとにカスタマイズされたサービスの導入を許容することが望ましい。このとき、個別の機能追加や機能変更によってシステムのある範囲ごとに異なる機能が提供されることになる。この範囲を「文化圏」と呼ぶ。「文化圏」は運用と改版の単位である。システム運用中になされる通信は文化圏内部でなされる場合が多く、この通信に問題は生じない。しかし、文化圏を越えた通信がなされた場合には、版の違いによる機能差異のために異常状態に陥る可能性がある。

3 リセプティブプラットホーム実現技術

本章では、2章で述べた問題を解決し、リセプティブプラットホームを実現するための諸技術について述べる。

分散アプリケーションモデル 従来のサーバクライアント型分散アプリケーションは、プロセス間の関係を二者間の主従関係に限定していた。しかし、大規模ネットワークを活用した通信システム、計算機ネットワークシステム、分散AI、CSCW等の分散アプリケーションは、自律的に動作、通信する多数のプロセスが複雑に関係する対等型分散アプリケーションである。これらはプロセス間依存関係に関して以下のような特徴を持つ。

- 同時に多数のプロセスと関係する。
- 関係するプロセスは動的に決定される。
- 互いにメッセージを送受しあい、対等な関係である。

したがって、ある変化の影響が波及する方向や範囲を容易に決定することが不可能であり、この決定の容易さに依存した従来手法[1),2)]の適用は本質的に困難である。リセプティブプラットホーム実現のための諸技術は、対等型分散アプリケーションに対して適用可能であることが必要である。

直接的影響への対処技術(変化発生支援技術):

(1) 改版実行のための技術: 過渡状態への対処によって、プロセスごとに独立した改版を実現する技術。これによって、サービスの継続提供を不可能にする複数プロセスの同期停止を回避する。

Receptive Platform – A System Evolution Environment
Hiroaki HIGAKI, Kenji MORIYASU, Hironobu OKUYAMA,
Yutaka HIRAKAWA, and Haruhisa ICHIKAWA
NTT Software Laboratories

間接的影響への対処技術 (波及影響対処技術): 改版による文化圏の発生を許容した場合に、文化圏を越えたインタラクションによる影響が波及する問題に対処する技術。

(2) **改版の影響がないことを保証する技術:** 既存サービスと新規サービスの仕様を解析し、文化圏を越えた通信を行なっても、新規仕様による動作が異常を生じない範囲にとどまっていることを確認する技術。

(3) **改版の影響を最小限にするための技術:** 通信可能な版の組み合わせを動的に検出することによって、文化圏を越えた通信を伴なうサービス提供を可能にする技術。これによって、ある文化圏で行なわれた改版に関する情報をその文化圏内に閉じることができるため、他文化圏に対する改版の影響を最小限に抑えることが可能となる。

以下では、特にプロセス間通信に関する異常状態 (未定義受信の発生など) を回避する技術について、(2) を設計段階での支援技術、(1) と (3) を運用段階での支援技術と分類して述べる。

3.1 設計段階での支援技術

文化圏を越えた通信に起因する異常動作を仕様の設計段階で検出、解消する技術について概要を述べる。

改版によりある文化圏に機能追加された場合を考える。この文化圏を越えて通信する場合、文化圏の独自機能 (既存機能には含まれない機能) を起動すれば異常動作することは自明である。しかし、独自機能を意図的に起動しない場合でも、勝手に起動してしまう場合がある。このような現象は、既存機能によって送信された信号を改版により追加された独自機能によって受信する場合に起こる。この信号の誤受信は、仕様中の特定動作パターンに起因している。したがって、誤受信の存在の判定、異常動作の検出が可能である[5]。

3.2 運用段階での支援技術

改版実行技術として、改版処理による未定義受信の発生を複数プロセスの同期停止によって避ける手法が提案されている[1), 2)]。しかし、対等型分散アプリケーションでは、複雑なプロセス間依存関係のために停止可能状態の獲得が困難であり、同期停止対象プロセス集合が一般に大きくなるという問題がある。この問題を回避し、かつサービスの継続提供が可能な改版処理[4]の概略は以下の通りである。

1. 文化圏内の各システム構成プロセスごとに、他の構成プロセスとは独立に新版を起動する。このとき、新版と同一の状態遷移が可能である限り旧版を並行動作させることによって、過渡状態 (旧版のみの構成プロセスと、新旧両版が並行動作している構成プロセスが混在した状態) が形成される。他の構成プロセスへのメッセージ送信は新版のみが行なう。
2. 文化圏内の全ての構成プロセスで新版が起動された場合には、旧版を停止させて改版処理を終了する。
3. プロセス間通信に未定義受信を生じ、正常なシステム動作を継続できない場合には、適切な状態にロールバックして旧版に処理を継続させる。異常状態に陥ることのないロールバックポイントは、旧版が新版の送受するメッセージを観測することによって決定可能である。

一方、文化圏を越えた通信によってシステムが異常状態に陥ることを回避する技術として、従来型分散アプリケーション実行環境では、通信可能な版間関係を集中的に管理し、起動時に版に関する合意を取る方法がある。しかし、大規模ネットワークにおいては版間関係の集中管理は不可能である。また、対等型分散アプリケーションではプロセス間依存関係は動的に決定されるため、起動時の合意が実行終了まで有効であるとは限らない。この問題は、上述した新旧版の並行動作と通信制御の応用によって、通信可能な版を動的に選択することで解決できる。

4 リセプティブプラットホームの適用

3.2節で述べた運用段階での支援技術の適用例を示す (図 2)。

端末で入力されたテキストを、ネットワークを介してファクシミリで送信するサービスを考える。サービス導入時はネットワーク全体で同一のサービスが提供されていたが、システム運用中にネットワークの一部で使いやすいように変更される場合がある。例えば、ファクシミリヘッダ用紙を付けるといった変更が考えられる。このとき、従来の処理にヘッダ用紙に記される情報のための処理が追加される。2 つの異なる版が同時にネットワーク上に存在した場合、どちらの版が提供されているかによって 2 つの文化圏が形成される。サービスが文化圏内で提供されている場合には問題は起こらない。しかし、自文化圏のファクシミリが故障で使用できないために、他文化圏のファクシミリを使用する場合には文化圏を越えた通信が行なわれる。このとき、両文化圏に共通の版 (例えば、初期導入版) が存在するならば、3.2節で述べた技術を適用できる。つまり、一方が新版を、他方が初期導入版を実行したために未定義受信などの異常状態が検出された場合には、共通の版 (ここでは初期導入版) を自動的に探策することによって、サービスを最後まで実行できる。

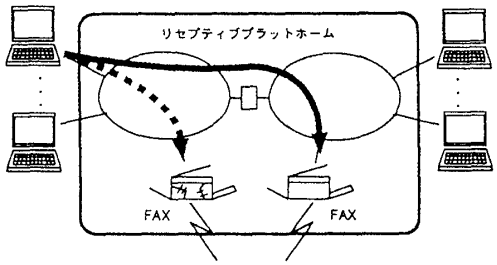

図 2: ファクシミリシステムへの適用

5 おわりに

本論文では、大規模分散システムの運用中に生じる様々な変化の影響を受容し、サービスを正しく安定的に提供する環境を実現するための分散プラットホーム、「リセプティブプラットホーム」を提案し、解決すべき問題と実現のための技術について述べた。リセプティブプラットホームの機能によって各ユーザは、異常状態に陥らないことが保証された環境でそれぞれ独立に多種多様なサービスを導入し、改良することが可能になる。これは、システムの進化過程と捉えることができる。つまり、リセプティブプラットホームは、システムの進化を支援する環境であるといえる。

今後は、リセプティブプラットホーム実現のための諸技術についての検討を更に深める予定である。

参 考 文 献

1) M.E.Segal and O.Frieder, "Dynamically Updating Distributed Software: Supporting Change in Uncertain and Mistrustful Environments," Proc. of IEEE Conf. on Software Maintenance, pp. 254–261 (1989).

2) J.Kramer and J.Magee, "The Evolving Philosophers Problem: Dynamic Change Management," IEEE Trans. Softw. Eng. , vol.16, No.11, pp. 1293–1306 (1990).

3) H.Higaki and T.Soneoka, "Fault-Tolerant Objects by Group-to-Group Communications in Distributed Systems," Proc. of 2nd IWRCS, pp. 62–71 (1992).

4) 桧垣, "分散システムにおける動的改版手法," 情報処理学会第 46 回全国大会, 1P–3, pp. 195–196 (1993).

5) 奥山他, "通信ソフトウェアの機能改良手法," 情報処理学会第 47 回全国大会, 6E–2 (1993).