能動オブジェクトシステムCAPEによるサーバ監視システムの動的負荷分散

4Q-1　　山口 実靖 *　　丸山 勝巳 **

* 東京大学大学院工学系研究科　sane@sail.t.u-tokyo.ac.jp

** 国立情報学研究所　maruyama@nii.ac.jp

1 はじめに

分散システムには大きく分けてClient-Server型(以下“C/S型”)とPeer to Peer型(以下“P-to-P型”)があるが,既存の分散プログラミングシステムの多くがRemote Procedure Call(以下“RPC”)に基づくC/S型をしている. しかし, 分散プロセス制御システムや通信制御システム等では各分散オブジェクトが並行動作し, 対等にメッセージを交換し合う. また, 相手の受理を待たずに自己の処理を続けなければならないことが多い. よって, このようなシステムの構築にはRPCベースのシステムでは不十分と考える. 上記の理由から我々は(1)P-to-P型で並行動作する, (2)非同期メッセージ交換が可能である, の2項を満たす分散オブジェクトライブラリCAPE[1]を実装した[1]. さらにCAPEを用いて並行動作型サーバ監視システムを試作し, その有効性を確認した[2].

本研究では, さらにサーバ監視システムを発展させ監視情報を用いた動的負荷分散を可能とした. また, 汎用フレームワーク化を一般サーバを監視すること, サーバおよび監視システムのチェックポイント処理も可能とした[2].

2 CAPE

CAPEは(1)P-to-P型並行動作, (2)非同期メッセージ交換, が可能な分散オブジェクトシステムライブラリでる. CAPEの実装はJava言語により行われており, Pure Javaであるため Java言語のプラットフォーム独立性が保たれている.

並行動作と能動オブジェクト　CAPEでは並行動作するオブジェクトを下記の能動オブジェクトとして定義する. 能動オブジェクトとは専用のスレッドを有し能動的に動作し, 専用のメッセージ行列を持つオブジェクトである. 能動オブジェクトではメッセージ転送と制御が分離されているため, 非同期メッセージ交換や高機能なメッセージ交換が可能である.

非同期メッセージ交換　CAPEでは非同期メッセージ交換が可能であり, メッセージ送信により送信元オブジェクトの並行動作が停止することを避けることができる. 送信元はメッセージを送信先の能動オブジェクトの待ち行列にキューイングだけ行い元の処理に戻るためブロックすることはない. メッセージを送られた能動オブジェクトは自分の待ち行列を自スレッドで順次処理していく(図1参照). 能動オブジェクトが送信元に戻り値を返す時は戻り値を予め指定されたReplyBox(戻り値入れ)に入れる. メッセージはプロセス間やネットワーク越しにも伝えることが可能であり分散環境でも用いることができる.

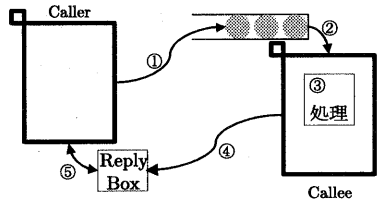

図1: CAPEにおける非同期メッセージの待ち行列

[1]Communicating Autonomous Programs Environment

[2]サーバのチェックポイント処理は, サーバがCAPEの能動オブジェクトを用いて実装されているときに限る. サーバ監視システムはつねにチェックポイント処理を行うことができる

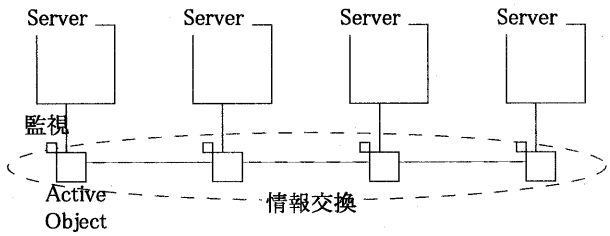

図2: CAPEを用いたサーバ監視システム

3 サーバ監視システム

提案システムCAPEを用いて並行動作しているサーバ群を監視するシステムを作成した. 図2の様に何らかの並行動作サーバ群を対象とし, 各サーバに監視用能動オブジェクトを起動する. 各監視用の能動オブジェクトは自ホストのサーバを監視しつつ, お互いにサーバの情報を交換する. 監視システムは各監視対象サーバにおいて並行に動作することが望まれ, P-to-P型が適していると考える. また, 監視システム間でのシーケンシャルな動作はほとんどないこと, 負荷情報等の交換は同期的に行われる必要はなく, 一方的に通知すればよいことなどから情報交換は非同期メッセージが適している.

システムの動作　監視用の能動オブジェクトは定期的に自ホストのサーバの状態(負荷状況など)を調査する. そして, 定期的に, あるいは状態が著しく変化したときに自ホストのサーバの状態を他ホストの監視用能動オブジェクト全てにマルチキャストする. これにより, 監視用の能動オブジェクトは常に新しい情報を保持することができる(完全な最新情報ではない).

動的負荷分散　提案システムは各サーバの負荷を等しくすることにより負荷分散を行う. 各監視システ

ムは自分の保持してる全サーバの負荷情報を元に完全に均等化がなされたときの負荷を計算する．すなわち，負荷の平均を求める (完全な最新平均値ではない)．次に，自ホストのサーバの負荷と平均の差を求め，超過分 / 不足分を求める．もし自サーバか超過している場合は，超過分を不足しているサーバに移動することを考える．移動量の合計は最大で超過分の半分である (不足しているサーバは逆に超過しているサーバから負荷を取り寄せるため)．負荷の移動は合計移動量を不足サーバの不足分で比例配分して行う．しかし，実際は合計移動量を最大量 (超過分の半分) とすると負荷が振動する過制御状態に陥ることがあるため，合計移動量を最大移動量より少なくすることが可能である (初期値は $\frac{1}{3}$ 倍)．

チェックポイント処理　一般のシステムでチェックポイント処理[3]を行うことは非常に困難なことであり，コストが必要とされることが多い．その理由は PCB(Process Control Block) 内の PC(Program Counter) およびスタック内のデータ (ローカル変数やリターンアドレスなど) の保存が容易でないことがあげられる．しかし，CAPE は能動オブジェクトに対するメッセージを自スレッドで処理するため，待ち行列内のあるメッセージを処理し終わり次のメッセージを処理する前はプログラムの実行中ではないためスタック内に情報が入っておらず，PC の保存も容易である (待ち行列内の次のメッセージを処理することから再開すればよい)．

4　実装と応用例

実装　前述の様に提案する監視システムはサーバに対して負荷情報を問い合わせ，負荷の移動 (除去と投入) を行うため監視されるサーバーはこのインターフェイスを実装している必要がある．監視システムはフレームワークとし実装されており，監視するサーバは監視システムの起動時にクラス名を文字列として指定する．

Java 言語で実装するサーバの監視　サーバを Java 言語を用いて実装する場合は監視が最も容易であり，サーバに指定されたインターフェイスを実装させればよい．起動時にはそのサーバのクラス名を指定すれば良い．

一般のサーバの監視　既存の (変更が不可能な) サーバを監視する場合はそのサーバに上記の要求を出すラッパを実装する必要がある．監視システム起動時には (サーバの代わりに) サーバのラッパを指定する．負荷の問い合わせ，移動のインターフェイスを提供していないサーバは扱うことが不可能であるが，この場合は不均一後の負荷均一化は本質的に不可能であるといえる．

評価用アプリケーション　評価用に簡単なデータベースシステムを実装した．A ～ D の 4 台が並列に動作する．サーバの負荷としては蓄積しているデータ量を用い，負荷に偏りが発生すると自動的に均等化が行われる．動作例を図 3に示す．例では 0 秒に A に 100 個，6 秒に C に 100 個，15 秒に D に 200 個の負荷 (データ) を追加したが，図 3の示されるように均等化が進んだ．ただし，監視周期は 1 秒，アナウンス周期は 10 秒，合計移動量は最大移動量の $\frac{1}{3}$ である．監視周期は評価実験でデータを入力する間隔より短くとる必要があり 1 秒とした．アナウンス周期は “大きな変化の検出によるアナウンスの実行” の効果も確認するために 10 秒と大きい値とした．

[3]実行中のイメージを保存すること

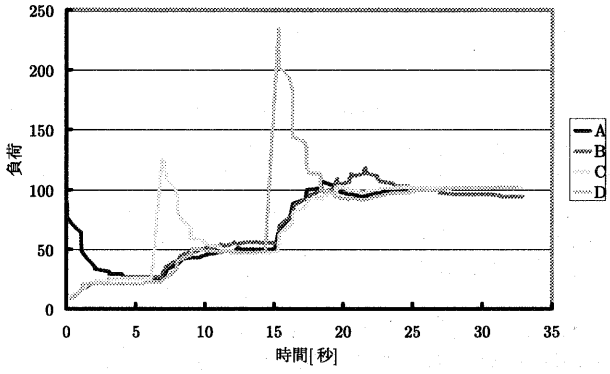

図 3: サーバ監視システム動作例

適用例　文献 [3] において述べられている “アイドル計算機を協調させて用いタスクを実行するシステム” において，各計算機の負荷状況を監視し負荷を均等化させる機能として使用した (監視されるサーバにラッパを実装する必要があった)．この適用例において各計算機のタスク数を均等化をすることが正しく行われ有用性が十分にあることが確認されたが，以下に示す問題も確認されさらなる改善が必要であると言える．(1) この例では移動要求が出されてから実際にタスクが移動されるまでの遅延が大きくアナウンス周期を長くする必要がある，(2) この例では負荷を完全に等しくすることよりも，過負荷状態の計算機を作らないことが重要である．

5　おわりに

本稿では (1)P-to-P 型並行動作，(2) 非同期メッセージ交換，が可能な分散オブジェクトシステムライブラリ CAPE について述べ，それによるサーバ監視システムを実装し説明をした．試作した評価アプリケーションにより監視システムの有効性を確認するとともに，今後の課題も確認された．今後の課題は，さらなる適用による実用性の確認，チェックポイント処理におけるファイル識別子の扱いなどである．

参考文献

[1] 丸山 勝巳, “分散能動オブジェクトシステムの Java ライブラリー CAPE”, 『学術情報センター紀要』第 12 号, 2000 年 3 月

[2] 山口 実靖, 丸山 勝巳, “能動オブジェクトシステム CAPE によるサーバ監視システム”, 第 61 回情報処理学会全国大会講演論文集 4D-3, 2000 年 10 月

[3] 山口 実靖, 相田 仁, 齊藤 忠夫, “協調バックグラウンドタスクスペースに関する検討”, 第 8 回マルチメディア通信と分散処理ワークショップ論文集 pp.85-90, 2000 年 12 月