

L-003

## グリッドへの Web アプリケーションの実装について An implementation of the web application on the grid

薄田 昌広†  
Masahiro Susukita

### 1 はじめに

グリッドコンピューティング技術は、ネットワーク上に分散した計算機資源を統合し、電力網のように必要なときに必要なパワーを利用できることを目指しており、高度な計算が必要とされる科学技術計算の分野において発展し、多くの成果をあげてきた。

しかし、一般事務アプリケーション分野においては、計算量が多いデータマイニング処理の一部など、限られた用途でのみ適用されている。

本研究では、一般事務アプリケーションについて、計算負荷の分散だけではなく、資源の効率的な利用および耐障害性の向上も含めてグリッドモデルを考察し、実際のアプリケーションをグリッドミドルウェア上に実装して動作を検証した。

### 2 Web アプリケーションのモデル検討

実際に社内で行われている Web アプリケーションの構成を調査したところ、ほとんどのアプリケーションが、Web サーバとデータベースから構成されており、Web サーバでの処理については、さらにユーザからの入出力を扱うユーザインターフェース部分(以下 UI 部)とユーザからの入力とデータベースの内容を元に業務ルールを実装したデータ処理部分(以下 DP 部)に大きく分かれている。UI 部は静的な HTML と入力を補助する JavaScript で記述され DP 部は Java サブレットで記述されている。(図 1)

Grid 技術を考慮せずに負荷分散を考えた場合、アプリケーション層スイッチを利用してクライアントの IP アドレスなどでアクセス先の Web サーバを振り分けることが考えられるが、このままでは特定の Web サーバの負荷が偏って増加することが考えられる。

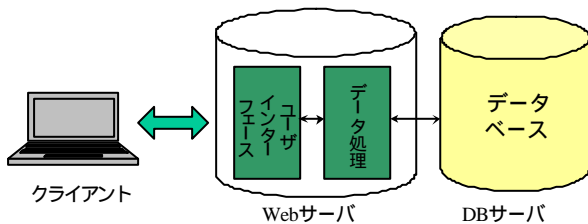


図 1 : アプリケーション構造

Web サーバ間での負荷分散については、UI 部がサーバ負荷に与える影響は少ないため、DP 部についてのみ分散処理を検討してグリッド上に実装する。ただし、耐障害性の面からは UI 部の分散も必要であるが、この点は従来のアプリケーション層・スイッチが利用でき、全体として図 2 のようなモデルを提案する。

† 関西電力株式会社

### 3 アプリケーションとミドルウェアの選択

このモデルを実装する対象の Web アプリケーションとしては、一般的な事務アプリケーションであるグループウェアを選択した。具体的には、JAVA で記述され、ソースが公開されているグループウェアである Group Session<sup>(2)</sup>を選択した。Group Session は小規模ではあるが、スケジュール管理や掲示板など一般的なグループウェアの機能をひととおり揃えている。

また、グリッドミドルウェアについては、もっとも標準的なものとして、世界的に標準化が進められている Globus Tool Kit<sup>(3)</sup>の Ver.3(以下 GT3)を選択した。GT3 も JAVA のアプリケーションとして動作するため、現状のシステムを変更することなく導入が可能である。

### 4 実装

Group Session の各機能を分析すると、ほとんどの場合においては個々の計算量が小さいが、アドレス帳検索機能については、登録されている全アドレスに対して検索処理を行う必要があるため、アドレスの増加とともに計算負荷が増大する。

検索処理は、同じ条件で大量のデータにマッチングをかけるため、検索の領域を分割して部分検索することで分散処理が可能である。そこで、部分検索処理を個別のジョブとして分離して他のサーバに再分配できるようなプログラム構造とした。そしてグリッドミドルウェアを介して他のサーバにジョブを分配し、処理結果を集約することで分散処理を実現する(図 2)。これによって計算機資源が効率よく利用され、全体の性能も増加することが予想される。

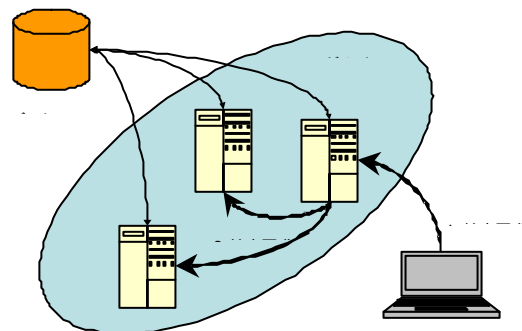


図 2 : 検索処理の分離

具体的な実装としては、Java アプリケーションサーバである Tomcat 上で Group Session および GT3 を動作させ、Group Session の検索モジュールのみ GT3 を介して部分検索処理への分散を行う(図 3)。

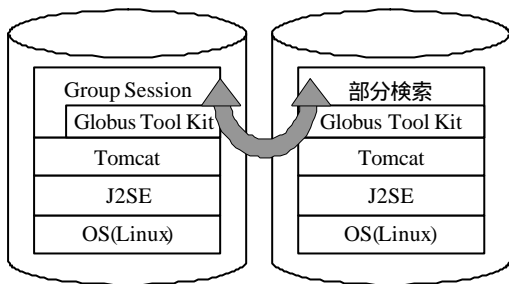


図3：アプリケーションの実装

平均処理時間 (msec)	実行ジョブ数		
	サーバA	サーバB	サーバC
9,015	60	26	26
9,697	84	--	28
9,921	77	35	--
9,995	112	--	--
12,881	--	58	54
21,812	--	112	--
23,572	--	--	112

表1：処理時間

## 5 検証

### 5.1 検証環境

性能の異なる PC サーバ 3 台に OS、グリッドミドルウェアおよびアプリケーションをインストールし、その間を 100Mbps の Fast Ethernet で接続してグリッドを構成した。また、Web クライアントは別ネットワークからルータを介して接続した(図 4)。

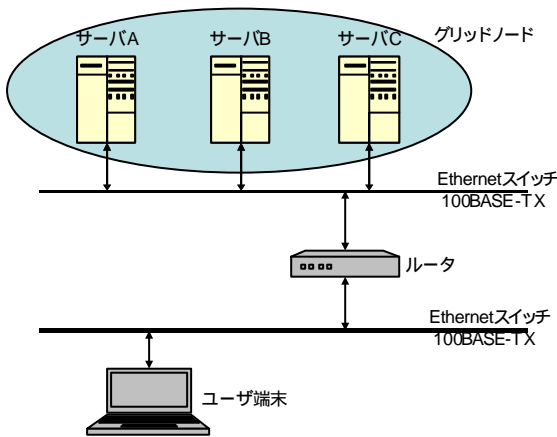


図4：検証環境

### 5.2 検証項目

検証項目としては、負荷の分散および耐障害性の二つの観点での検証を実施した。具体的には、1 台から 3 台までの PC でアプリケーションを稼働させ、処理時間と各 PC での処理ジョブ数を計測した。また、処理の途中で 1 台の PC をネットワークから強制的に切り離すことで障害を模擬し、その場合でも正しい結果が得られることを確認することとした。

### 5.3 検証結果

負荷分散試験の結果は表 1 のとおりであり、台数に応じた処理時間がかかり、性能に応じて実行ジョブが割り振られている。ただし一部の結果については台数を増やしても処理時間がほとんど変わらない例があった。これについてはグリッドミドルウェアによるオーバーヘッドが無視できないためと思われる。

また、耐障害性試験の結果、処理時間は通常よりも長くはなったが正しい検索結果が得られ、耐障害性機能が正常に動作していたことがわかった。

## 6 まとめ

一般的な事務アプリケーションの代表例として Web アプリケーションをグリッドミドルウェア上に実装し、動作を検証することでグリッドコンピューティングの一般事務アプリケーションへの適用の有効性を確認した。

しかし、負荷を分散して台数を増やした場合に、必ずしも全体の処理性能が向上するわけではなく、ミドルウェアを介することによるオーバーヘッドが小さくないこともわかった。

今後は、より規模の大きい一般事務アプリケーションへの適用と、グリッドミドルウェアによるオーバーヘッドをなるべく少なくするための実装を検討する必要がある。

## 参考文献

- (1) Ian Foster, Carl Kesselman : “The GRID Blueprint for a New Computing Infrastructure” (1999)
- (2) Group Session : <http://www.gs.sjts.co.jp/>
- (3) Globus Toolkit : <http://www.globus.org/>