新たな DNAS の開発

Development of new DNAS

谷口 義樹
Yoshiki Taniguchi

Abstract: Recently, the Grid has been paid attention and many Grid Middleware (Globus, DNAS, GMWS, Jojo and so on) has been developed. Distributed Network Application System is a dynamic tree structure system for the Grid. Grid Master Worker System is a master-worker system for the Grid. Jojo is a java-based programming environment for the Grid. Each system has some advantages and some disadvantages. In this paper, a new middleware system for the Grid, which considering their advantages and disadvantages, is proposed.

1　はじめに

近年，広域に分散した計算資源や情報資源をネットワークで接続し，仮想的な一つの環境として統合して利用可能なグリッドが注目を集めている．また，Globus Toolkit に代表されるようにグリッドを利用してアプリケーションを開発するためのミドルウェア，ライブラリが数多く研究・開発されている．本研究室でも，数年前より Distributed Network Application System（以下，DNAS）や Grid Master Worker System（以下，GMWS）と呼ばれるグリッド等の広域分散環境に適応可能なミドルウェアの開発を行ってきた．本研究では，これまで本研究室で開発されてきた DNAS および GMWS の利点と欠点を明確にし，利点を残しながら欠点を取り除くように新しい DNAS の設計目標を示し，DNAS の再開発を行うことを目的としている．

2　既存ミドルウェアの特徴

2.1　DNAS

DNAS は，上川らによって開発された動的に構成を変えることのできる P2P 志向のツリー型ミドルウェアである．このシステムは DNAS master と DNAS servent の 2 つのデーモンプログラムから構成される．DNAS master はツリー構造のルートノードで動作するデーモンプログラムであり，1 つの環境毎に唯一の存在である．DNAS servent は DNAS master 以外の計算ノードで動作するデーモンプログラムのことを指し，実際にアプリケーションは DNAS servent が動いているノードでのみ実行される．各 DNAS servent ノードで動作するアプリケーションは，ローカルホストの DNAS servent デーモンに情報を送り，デーモンプログラムが一定時間毎にその情報を上位のノードに伝達するといった動作を繰り返すことで，各ノード間で情報の交換を行うことが出来る仕組みになっている．情報は基本的に下から上へ送られることになる．DNAS は DNAS API を提供しており，これを利用することで DNAS の通信を用いたアプリケーションの開発が可能である．DNAS の動作する様子を Fig. 1 に示す．

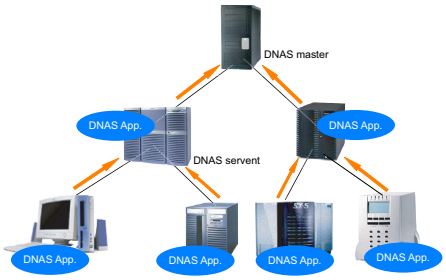

Fig. 1　DNAS の動作

DNAS の最大の特徴は，動的なトポロジの自動再構成機能である．DNAS のトポロジの再構成の様子を Fig. 2 に示す．通常，ツリー構造のシステムは，中間層のノードが障害などによって離脱してしまうとその下に接続していたノード全てがシステムから切り離されてしまう．しかし，DNAS では障害発生時に下位ノードの接続先を変えるなどしてその問題を回避することができる．またある特定のノードに対して多大な負荷が掛かることを防ぐために，下位ノード数を制限することで負荷分散を実現している．

一方，DNAS の短所として挙げられるのは，情報の流れが基本的に下から上への一方向のみであるため情報に偏りが生じる点，セキュリティ機構が不十分である点が挙げられる．また DNAS master という唯一特別なプログラムが絶対的に常に動いている必要がある点が挙げられる．

2.2　GMWS

GMWS は，谷村らによって開発されたマスタワーカ型のグリッドミドルウェアである．それぞれの計算資源においてデーモンとして常駐し，GMWS アプリケーショ

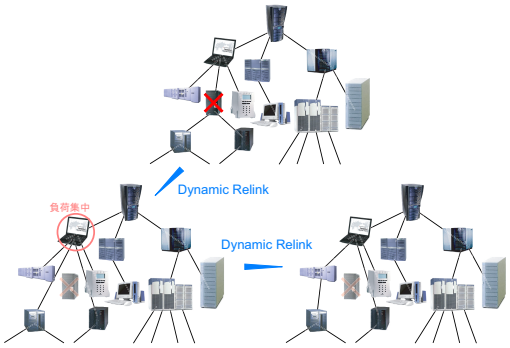

Fig. 2　DNAS の動的なトポロジの再構成

ンの実行をサポートする Agent プログラムが存在し，唯一のマスター Agent とそれによって管理されるワーカ Agent の 2 種類の Agent が存在する．マスター Agent の役割は，その下で働くワーカ Agent の起動，ワーカ Agent へのジョブの割り当て，ワーカ間の通信の仲介である．ワーカ Agent はマスター Agent の指示に従って動作し，ワーカ Agent 同士はお互いの存在を知らず，通信も行わないという仕組みになっている．GMWS を利用する際には，あらかじめマスター Agent に利用資源の情報を記述したファイルを与える．GMWS の動作の様子を Fig. 3 に示す．

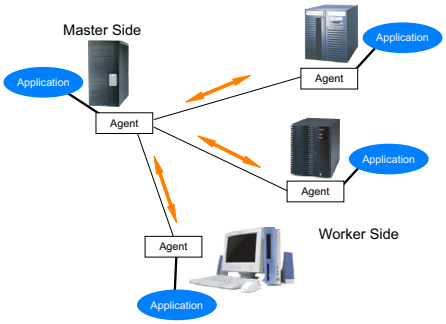

Fig. 3　GMWS の動作

GMWS はマスター Agent からのワーカ Agent の起動，各計算資源間の通信に Globus を利用することができる．これにより，SSH を利用したものより強固なセキュリティ機構を有する．GMWS の短所は，利用するワーカの台数をあらかじめ決定しておく必要があるため動的なノードの増減に対応できない点，またワーカの構成が全てフラットなため，ノード数が増えるにつれてワーカへの負担が大きくなる点が挙げられる．

2.3　Jojo

Jojo は産業技術総合研究所の中田らによって開発された，Java を用いて実装された階層構造を持つ環境に適した分散実行環境である．Jojo の主な特徴は，Globus や SSH を用いたセキュアな起動と通信，複数の種類のメッセージパッシング API，プログラムコードの自動アップロードである．Jojo は，システムがデーモンプログラムとして各ノードに常駐するのではなく，一種の通信ライブラリであるため，動的な環境の変化には現在のところ対応していない．

3　DNAS2 の設計

第 2 章で述べたように，既存のミドルウェアにはそれぞれ長所と短所が存在している．これらを参考に，各ミドルウェアの特徴を活かした新たなミドルウェアとして DNAS2 の設計を行う．以下にその設計目標を示す．

- プラットフォームに依存しない Java 言語を用いて実装する．
- 特定のノードに負荷が集中しないように，システムの形態をツリートポロジーとする．
- ノード間の情報伝達は接続されているデーモンプログラム同士によって定期的に行われる．また，アプリケーションのための簡易インターフェイス (API) を提供する．
- アプリケーション実行中の動的な環境の変化（ノード障害や負荷の集中，ノードの追加）にシステムが動的に対応できる．そのためにライブラリ型ではなく，デーモンプログラムとして常駐させるようにする．
- DNAS2 が動作する環境内の全てのノードにおいて，同じデーモンプログラムが動作する（唯一の特別なデーモンプログラムを作らない）．
- 負荷の状況に応じてアプリケーションが実行を一時停止したり，実行を再開できるような情報をシステム側からアプリケーションに提供し，それを簡単に利用することが出来る．
- SSH や Globus を利用した強固なセキュリティ機構を有する．
- 各ノードの性能に応じて，与えるジョブの大きさを変えられるような仕組みを提供する．
- 実行ファイルの配布の仕組みを有する．

4　まとめと今後の課題

本研究では，既存のミドルウェアの長所と短所を明確にし，それらを組み合わせた新しいミドルウェア DNAS2 の開発を行っている．現在のところ，デーモンプログラムおよびアプリケーションへのインターフェイス，動的なトポロジー変化機能，負荷状況の通知機能の実装が完了している．今後の課題として，より強固なセキュリティ機構，ジョブの大きさを可変にする仕組み，実行ファイルの配布の仕組みを実装することが挙げられる．また，本システムを利用した有効なアプリケーションを検討する必要があると考えられる．