マルチメディア通信と分散処理 112−11
（２００３．３．２０）

災害時にも有効な疎結合分散システム用汎用トランスポートシステム

井澤 志充, 大島 龍之介

インターネット応用技術研究所

我々は、IAA システムで用いられているトランスポート部を元に、さらに汎用化し、耐規模性と処理速度の向上を図った、疎結合分散システム用汎用トランスポートシステムとして KANI を設計した。そこで本稿では、KANI の設計コンセプト、基本設計とその実装について述べる。また、従来の IAA トランスポートの抱えていた課題をどのように解決しているかについて述べる。次に、KANI の実用性について、シミュレーションの結果を用いて示す。さらに、2002 年 9 月 1 日に行われた、練馬区・東京都合同防災訓練にて、IAA システムのうち、IAA トランスポート部を KANI と交換して、実際に運用実験を行った。この公開実験によってえられた知見についても報告する。

A wide purpose transport system for the loosely coupled widely distributed system which is used in a disaster case effectively

Yukimitsu Izawa, Ryunosuke Ohshima

Institute of Applied Internet technology, Inc.

We design a KANI transport system which is a wide purpose transport system for the loosely coupled widely distributed system. KANI is based on IAA system's transport system for multi-purpose, more scalable and improvement in speed. This report describes a design concept, basic design and implementation of KANI. In next section, We describe enhancement of KANI for IAA system transport. In addition, we desctribe utility of KANI with explanation with simulations. At last, we report experiment of KANI in the actual use.

1 はじめに

従来、広域に分散したシステム間のデータ通信をサポートするデータ配送機構は、安定したネットワークを前提にしているものが多く、安定して稼働するサーバがネットワーク上を移動するシステムのサポートを行うなど、何らかの安定したシステムを前提としたモデルで設計されている。

このようなモデルで設計されたデータ配送機構を用いる問題点として、大規模災害時に発生するようなネットワークの激しい輻輳や分断が

起こった場合などを想定していないため、そのような場合には利用することができない。

これは従来のデータ配送機構が、配送における頑健性を念頭に置いて設計されていないことに起因している。つまり、配送における頑健性を提供することができるデータ配送機構であれば、大規模災害時でも有効に機能することができるといえる。

本稿で提案する KANI は、災害時におけるネットワークの分断や遅延の発生を前提とし、これらの問題に対して Best effort 型のデータ配送を行うシステムである。

次節では、まず KANI の基本設計について述べる。

2 設計コンセプト

本稿で提案する手法のねらいは、以下のような要求をもつデータ通信における信頼性を向上させることである。

- 広域に分散したシステム間での同報データ通信
- 即時性は問わない
- 耐規模性に優れる
- できるだけ汎用的であること。
- 災害時の様な緊急時の運用に柔軟であること。
- 誰にでも使えること。
- セキュアであること。
- 移植性が高いこと。

1 番目の項目における同報データ通信とは、データ配送機構によって接続されている分散したシステム全てに同様の情報を伝搬する機能を有する必要があることをさしている。これは、1 対多での通信をサポートすることを意味する。1 対多通信をサポートすることで、例えばこのデータ配送機能によって結ばれる分散データベースの全てのクラスタで同様のレコードを容易に保持することができるようになり、分散データベース全体の系に冗長性を持たせることができる。2 番目の項目としてデータ配送の即時性を重要視しないことを挙げた。これは、即時性と頑健性がトレードオフの関係に陥り易いことに起因しており、我々は即時性よりも頑健性を重要視しているためである。3 番目の項目として挙げた耐規模性は、このデータ通信機構が様々なアプリケーションが利用できるようにするためには重要な要素である。KANI は IAA システム [1] の一部として設計されてきた経緯から、災害時のようにネットワークの状態が不安定でも出来る限りデータの配送に努力するように設計した。

また、KANI は分散ソフトウェア環境を提供し、IAA システムはその上で動く、一アプリケーションであるように分離して設計されており、KANI の提供する分散網を利用する全く別のアプリケーションも容易に構築できるようにした。

KANI は、分散網へのクラスタの追加投入も視野にいれた動的な網制御機能を持っており、配送網自体もネットワークの状況に応じて最適と思われる経路でデータ配送を行う。

また、KANI の通信はすべて SSL を用いた暗号化通信路を使って通信が行われる。これは、IAA システムで用いる安否情報などの個人のプライバシデータが、安易に漏洩することを防ぐとともに、汎用なデータ配送網としては必須の要件である。

そのほか、複雑な設定を行わずに運用できるような設計を心掛け、設定ファイルはひとつのディレクトリに集め、簡便な記述フォーマットを心掛けた。これは、緊急の場合に誰でも使えるようにするためには重要な要素である。

現在、このような要求をデータ配送機構のみで満たすものはみられず、データ配送機構を用いるアプリケーション側で必要に応じた処理を行う必要がある。

そこで本稿では、データ配送機構に広域負荷分散や同報性といった機能を拡張できるような枠組みを持たせ、データ配送において要求される機能をデータ配送機構に組み込むことが出来るシステム設計を提案する。

3 基本設計

前節で述べたように、従来のデータ配送機構と比較して高い拡張性や耐規模性、頑健性を提供するため本稿で提案するデータ配送機構には、大別して 4 層からなるレイヤモデルを採用している。

本節では設計のうち特に本方式のレイヤ構成について述べる。

3.1 レイヤ構造

本稿で提案するデータ配送機構は、端点に位置するシステム同士の通信 (point-to-point) から、グループ化されたデータベースクラスタへのデータ送信まで、様々なデータ通信の要求を満たすことができるように設計されている。

またアプリケーションには仮想的なホストを提供し、これらとのデータ通信を行わせる方式を採用することで、アプリケーションに対して実際のネットワークトポロジーを隠蔽することができるようにした。

これは、配送ネットワークの再構成やグループ化、あるいは激しい輻輳や分断といったネットワーク故障が起こった場合の自動対応などをアプリケーションに意識させる事なく行えることを意味している。あるいはアプリケーションがこれらの機能を明示的に指定することも可能である。

このような粒度の違う要求をみたすため、本稿で提案するデータ配送機構に以下の 4 つのレイヤを定義した。この 4 つのレイヤは以下のように構成されている。

- リンクアソシエーション層
- リンクコンフィギュレーション層
- レコード層
- レコードコントロール層

これら 4 層からなるデータ配送機構の上位層にはアプリケーションレイヤがあり、これにサービスを行う。また下位層には TCP といったセッション指向のプロトコルの他にも UDP のようなデータグラム指向のプロトコルも使用することができる。

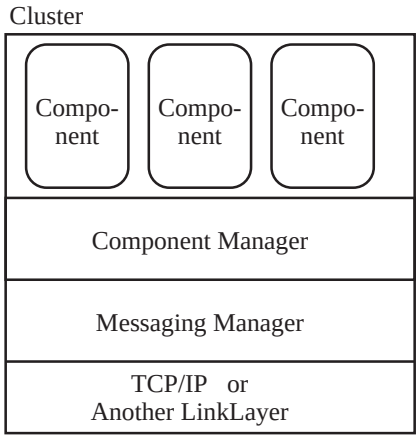

図 1: コンポーネントモデル

これら 4 つのレイヤモデルについての詳細は、[2] を参照されたい。

3.2 コンポーネントモデル

今回の実装では、KANI 上のアプリケーションを機能毎に分割し、異なるアプリケーション間で部分機能を共有できる、コンポーネントモデルを導入した。

これは、KANI 上にコンポーネントと呼ばれるソフトウェアモジュールを組みたてていき、これを組み合わせることでひとつのアプリケーションを構成する、ビルディングブロック方式のモデルである。

これにより、アプリケーションの開発コストの軽減と、既存の KANI アプリケーションの再利用が可能となる。

KANI の実装では、このコンポーネントを管理するコンポーネントマネージャと、データ配送管理を行うメッセージングマネージャから構成されており、この上でコンポーネントが動くことになる。

これを図に表したのが、図 1である。

このうち、前節の 4 つのレイヤ構造を実現しているのはメッセージングマネージャである。

コンポーネントは、サービスの単位毎に分離

して作成することで再利用性を考慮し、様々なアプリケーションに再利用されることが可能である。アプリケーションはこれらのコンポーネントの集合であるといえる。

次にコンポーネントマネージャについて説明する。コンポーネントマネージャは、コンポーネントを統合するのが仕事であり、以下の役割をになっている。

- ローカルコンポーネントの実行主体
- ローカルコンポーネント間の通信のサポート
- リモートコンポーネント呼び出しのサポート

コンポーネントマネージャが仲介する全てのコンポーネント間の通信は、メッセージのやりとりとして抽象化されており、ローカルにあるコンポーネントとの通信とリモートにあるコンポーネントとの通信は、その違いを意識すること無く利用できるようになっている。

次にメッセージングマネージャについて説明する。メッセージングマネージャは、クラスタ間の通信を担うのが仕事であり、コンポーネントマネージャから見た場合にはコンポーネントのひとつである。

メッセージングマネージャは、IP アドレスなどのレイヤ 3 情報を隠蔽し、コンポーネントマネージャにはクラスタ内部での識別子であるノード ID を情報として渡している。これによって、レイヤ 3 が IP 網以外である場合でもメッセージングマネージャの変更のみで対応できるようになっている。

メッセージングマネージャはデータの配送に関して、以下のようなプライオリティで設計されている。

1. 可能な限りデータを届ける努力をする
2. 可能な限り早く伝達させる
3. 可能な限りトラフィックを減らす

つまりメッセージングマネージャは、データの配送の確実さを、伝達速度よりも優先する。次にメッセージングマネージャのデータ配送方法について説明する。

メッセージングマネージャは、各クラスタとフルメッシュの配送網を構築する。各クラスタは、プライオリティ付き出力キューを持っており、プライオリティが高いほど、頻繁にデータの送信を試みる。このプライオリティは自動的に変更されるようになっている。クラスタ間のデータのやりとりは、最初にデータの ID を相手に持っているかどうかを問い合わせる、いわゆる Ihave/Sendme 方式をもちいている。プライオリティは、相手がすでにデータを持っている場合には、プライオリティは下がるように、相手が持っていない場合はプライオリティがあがるようになっている。プライオリティが低くても必ずデータの送信は試みるようになっており、どこかひとつのクラスタと接続性が確保されていればデータは届くようになっている。

これにより、できるだけ確実に早く、さらにトラフィックをできるだけ抑えたデータ配送を行っている。

4 配送戦略

今回、我々が実装した KANI の配送戦略について説明する。メッセージングマネージャは、データの配送に関して、以下のようなプライオリティで設計されている。

1. 可能な限りデータを届ける努力をする
2. 可能な限り早く伝達させる
3. 可能な限りトラフィックを減らす

つまりメッセージングマネージャは、データの配送の確実さを、伝達速度よりも優先する。次にメッセージングマネージャのデータ配送方法について説明する。メッセージングマネージャは、各クラスタとフルメッシュの配送網を構築する。各クラスタは、プライオリティ付き出力キューを持っており、プライオリティが高いほど、頻繁にデータの送信を試みる。このプライオリティは自動的に変更されるようになってい

る。クラスタ間のデータのやりとりは、最初にデータの ID を相手に持っているかどうかを問い合わせる、いわゆる Ihave/Sendme 方式をもちいている。プライオリティは、相手がすでにデータを持っている場合には、プライオリティは下がるように、相手が持っていない場合はプライオリティがあがるようになっている。プライオリティが低くても必ずデータの送信は試みるようになっており、どこかひとつのクラスタと接続性が確保されていればデータは届くようになっている。

これにより、できるだけ確実に早く、さらにトラフィックをできるだけ抑えたデータ配送を行っている。

現在の配送キューの管理は、あるクラスタの出力キューを見た場合に、それぞれのキューのフラッシュ間隔は、$BaseInterval * 2^n sec$ で表すことができる。n はプライオリティを示している。つまり、$BaseInterval = 0.5sec$ の場合には、プライオリティの高い順に、$1, 2, 4, 8, 16 \cdots$ と出力キューのフラッシュ間隔は設定される。

出力キューのデータのフラッシュは、Ihave/Sendme 方式の問い合わせを行い、その結果によってキューのプライオリティを動的に変更する。つまり、Ihave と返ってくる数が多いキューよりも Sendme と返ってくる数の多いキューのプライオリティを上げるという操作を行う。これにより、データ配送における有効なパスのプライオリティをあげることになる。

5 シミュレーション

前述の配送に関するアルゴリズムの有効性をシミュレーションによって示す。まず、理想的な配送における配送にかかる時間を示す。理想的な配送とは、全てのキュー出力に対して、Sendme 応答しか返らず、全てのクラスタのプライオリティが理想状態であることを指す。

この場合に配送にかかる時間を表したのが図2である。 X 軸はデータがある一つのクラスタに投入されてから全てのクラスタに行き渡るまでの時間をあらわす。Y 軸はクラスタの数をあらわす。なお Y 軸は対数表現である。これをみるとわかるように、100 程度のクラスタ構成であれば、$8BaseInterbal$ あれば全てのクラスタにデータが行き渡らせることができる (ネットワークの遅延時間を 0 とした場合)。

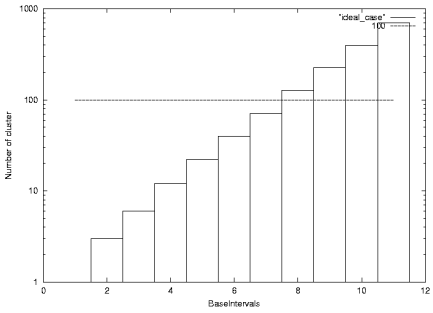

図 2: 理想的な配送における配送時間

つぎにワーストケースにおける配送時間を計算する。ワーストケースとは、本来はフルメッシュであるはずの配送網が、プライオリティによって見掛け上、全てのクラスタが一列に並んだ状態である。これによって、すべての配送は最初の BaseInterval 経過後、つまり最も高いプライオリティのキューフラッシュによってのみ高々一つのクラスタにデータを配送することになる。

従って、ワーストケースにおける配送時間は、$BaseInterval * 2^1 *$ クラスタ数 (sec) となる。つまり、クラスタ数が 100 で、$BaseInterval$ が 0.5sec の場合には、全てのクラスタに配送されるのに 100sec かかる計算になる。

これは、非常に稀なケースであり、また初期状態のクラスタの構成をある程度手動でおこなうことによって、用意に回避できると考える。

6 従来の IAA トランスポートと比較した場合の優位点

従来の IAA システムで用いていたトランスポートシステム [3] と KANI の比較を以下に示す。

配送網のうち動的な評価ルーチンによって優先配送を行うようになった。これにより、網の現状に即した配送経路を選択し、かつバックアップの配送もおこなえるようになり、より配送に関する堅牢性が増したといえる。

また、データの配送においてはSSLを用いて配送するようにし、従来のDES暗号方式を撤廃した。クラスタ毎に鍵を持たせ、その鍵に基づいた公開鍵暗号方式を用いたデータの暗号化をおこなった。これにより、全てのクラスタでひとつの秘密鍵を共有する必要がなくなった。

従来のIAAトランスポートは配送するデータはテキストデータを対象とし、データ構造を持ったままの配送は不可能であったが、KANIでは、Javaのシリアライザブルデータのストリーミング機能をもちいることで、アプリケーションのもつデータ構造を保持したままデータ転送できるようになった。これによりKANIはより汎用なデータ配送メカニズムとして利用できることを意味している。

また、実装言語を従来のPerlからJavaに移行することで、セキュリティ面での配慮や開発行程の短縮、より高いポータビリティを確保した。従来のPerl版は、Perlモジュールに高く依存しており、各種モジュールのバージョンによっては、動作に不具合が生じることがあった。

またコンポーネントモデルを採用し、アプリケーションと配送部分を分離することで、IAA以外のアプリケーションの開発が容易になった。

7 実運用によって得られた知見

われわれは、2002年9月1日に行われた、練馬区・東京都合同防災訓練にて、IAAシステムを運用した。このIAAシステムのトランスポート部をKANIに入れ替え、実際に運用実験を行った。

当日は被災者情報として162件の登録があったが、これらは全て1秒以内にデータ交換され、データの欠損もなかった。また、全てのデータがもつデータ構造は配送先で正しく再構成され、コンポーネント間の通信が確立されたことを確認した。

運用の課題としては、SSLを使った通信を行う際の鍵交換を、暗号メールベースで行ったが、実際にKANIが使う鍵として組み込む作業は手作業で行っており、作業に不馴れなものが扱うとクラスタの立ち上げに手間取ることがわかった。

8 おわりに

本稿では、KANIの設計コンセプトおよび基本設計について述べ、コンポーネントモデルによってアプリケーションに対して、容易に利用可能な疎結合分散環境を提供できることを示した。また、KANIは災害時などネットワークが不安定であるような環境下での運用を設計の段階で想定しており、災害時の運用においても頑健であることを述べた。また、KANIのデータ配送戦略について述べ、BestEffor型の配送を行うことで、データの配送は速度よりも信頼性を重視していることを述べた。また、上記の戦略を用いた場合の有効性について、シミュレーションの結果を持って示した。次に 従来のIAAトランスポートとの比較を行い、最後に実運用によって得られた知見についても述べた。

参考文献

[1] 井澤志充, 木本雅彦, 多田信彦, 大野浩之, 篠田陽一. Iaaシステムの現状とその課題. pp. 15–27, November 2000.

[2] 井澤志充, 三輪信介, 篠田陽一. 広域疎結合分散システムのためのデータ配送機構の設計. 情報処理学会 マルチメディア通信と分散処理ワークショップ論文集 ISSN 1344–0640, pp. 67–72, December 1999.

[3] 井澤志充. NetNewsを使った信頼性のあるデータ通信の技法. 情報処理学会 分散システム運用技術 研究報告 No.9, pp. 49–54, May 1998.