Vol.2009-DPS-139 No.12
2009/6/19

Content-Based Network におけるデータ処理コンポーネント最適配置問題

落合秀也[†1] 木村慎吾[†2] 松浦知史[†3]
砂原秀樹[†4] 福原英之[†5] 江崎浩[†1]

分散 Pub/Sub システムの一形態である Content-Based Network(CBN) において，定期的に配信されるデータを購読し，解釈後，意味のある情報を生成し，再配信するサービス (データ処理コンポーネント) を考える．データソースから情報のエンドユーザまでのデータ配送トポロジを最適化する動的サービス配置について基礎的なモデル化を行い，想定される実装方法について提示する．

Locating Data Processing Components on Content-Based Networks

Hideya Ochiai,[†1] Shingo Kimura,[†2]
Satoshi Matsuura,[†3] Hideki Sunahara,[†4]
Hideyuki Fukuhara[†5] and Hiroshi Esaki [†1]

We add data processing components into a content-based network(CBN). A CBN is a distributed publish/subscribe system, which transparently delivers messages from publishers to subscribers. The components produce and publish meaningful information by subscribing and processing the data collected on the CBN. We present a framework to model the topology of data flow from the data sources to the information users, which should be necessary to calculate the optimized location of the components in the network.

1. はじめに

Content-Based Network(CBN)[4),5),11)] はメッセージの Publish/Subscribe 配信を分散的に実現するシステムであり，インターネット上にオーバレイネットワークとして実現されるケースが少なくない．広域的なセンサデータの配信 (共有)[7)]，コンテンツ配信，警戒情報の発令，ネットワーク対戦ゲームなど，様々な情報共有のためのプラットフォームとして有望なネットワークである．CBN は，Publish/Subscribe システムの中でも，特に Content-Based Publish/Subscribe システム[6)] に分類されるものであり，メッセージの内容単位で購読条件を指定し，ルーティングが行われる．CBN 自体は透過的なメッセージ配信プラットフォームであり，ネットワーク内でのデータ処理や加工は行わない．CBN にデータ処理機能を持たせれば，総合的な通信基盤としてより充実するが，システムとしての最適化を行わない限り，十分なパフォーマンスが得られないことがある．

本研究では，CBN にデータ処理機能を持たせることを考えた場合に，通信遅延の低減，データ処理負荷の低減，トラフィック量の削減といったネットワーク全体での最適化問題について扱う．具体的には，データ処理コンポーネントの実行場所をネットワーク内で適切に設定することで，これらの最適化を行うことを考える．本研究は，このような環境下での最適化を実現するための計算モデルを提案することを狙いとしており，具体的な最適化基準に対する最適化方法については，オープンな課題としている．

以下，データ処理機能の搭載された CBN の概要を述べる．データ (それ自身では大した意味を持たない) の配信ソースがたくさんある環境において，それらから公開されるデータを購読し，なんらかの加工処理を行うことで意味を持つ情報を抽出できるケースがある．例えば，広域に渡って設置されたセンサが観測する気温，湿度，気圧，風向風速，雨量などのデータから，どの場所で集中豪雨が発生しやすいかや，その雨がどの方角に向かっているかなどの情報を，統計的に導出することは可能だろう．このようにして得られた情報は，多く

†1 東京大学/NICT
The University of Tokyo/NICT
†2 奈良先端科学技術大学院大学
NAIST
†3 奈良先端科学技術大学院大学/NICT
NAIST/NICT
†4 慶應義塾大学/奈良先端科学技術大学院大学
Keio University/NAIST
†5 Net One Systems/Net One Systems

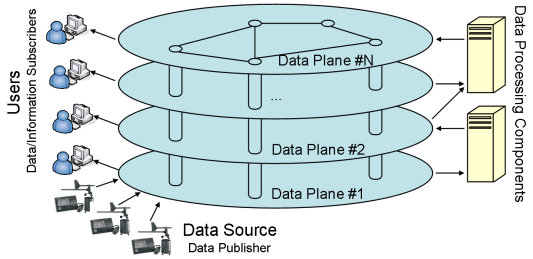

図 1 CBN プラットフォームでのデータ処理コンポーネント
Fig. 1 Data processing components on a content-based network

のユーザに関心があるため，CBN を使ってこれを再配信することは自然な考えである．その結果，次のような形態で CBN を運用することになる (図 1)．

N 種類のデータ・プレーン： それ自身では大した情報を持たないデータと，ユーザがそのまま利用することのできる情報，を配信する二つのデータ・プレーンを用意することを考えよう．前述の広域センサネットワークの例では，センサは前述のデータ・プレーンに対して観測データを公開し，ユーザは後述のデータ・プレーンから情報をもらう．一般に N 種類のデータ・プレーンが CBN の上で管理されうる．

データ処理コンポーネント： 前述のデータ・プレーンから処理に必要なデータを購読し，なんらかの加工処理を行う．そして，その結果得られた情報を，後述のデータ・プレーンを通じて発信する．

上述の仕組みによって，データ処理機能の搭載された CBN が完成する．本論文では，Publisher/Subscriber の他に，データソース/ユーザという用語を用いる場合がある．Publisher/Subscriber は，CBN において透過的なメッセージ交換を行う際の送信者/受信者を表すのに対し，データソース/ユーザは，データの流れの途中にデータ処理コンポーネントが含まれる場合の，末端の送信者/受信者にも使うことができる，より広範な表現である．

CBN は，インターネット上のオーバレイネットワークとして実装されるケースが多いため，実際には IP ネットワークでのパケット配送遅延に性能が拘束されるという弱点がある．特に，IP ネットワークでのパケット配送遅延が CBN での 1 ホップ辺り 100ms のオーダであった場合，末端のデータソースから，最終的なエンドユーザまでに，合計 10 ホップ必要なケースでは，1s の遅延が発生してしまう．データソースから最終的なユーザまでの間に，データ処理が何段も入る場合，10 ホップ程度では済まないケースも十分に考えられる．リアルタイム応答を期待するアプリケーションや，緊急性を要するアプリケーションに利用したい場合においては，この遅延はできる限り減らしたいものである．

さらには，トラフィックを全体的に減らしたい場合や，空いている計算リソースを有効に使用したい場合などもあるだろう．具体的な最適化基準は各種環境によって異なることを踏まえ，本論文では，より抽象的なレベルにおいて，最適化計算を行うための基礎的なモデル化を行うことを主眼としている．

なお，本研究の段階では，既存の CBN のルーティング管理は，データ処理コンポーネントの配置管理とは，独立なものとして扱うものとする．

本論文の構成は，次のようになっている．第 2 章で関連研究について述べる．第 3 章にて，データ処理コンポーネントの配置最適化のモデルを定義し，第 4 章にて想定される実装方法について整理する．第 5 章で結論を述べる．

2. 関 連 研 究

Content-Based Network(CBN)[4),5),11)] は，Content-Based Publish/Subscribe の通信方式[6)] を分散的に実現したものである．メッセージの送信者と受信者は，それぞれ Publisher，Subscriber と呼ばれ，Subscriber が購読したいメッセージの内容的な分類を Predicates によって指定する．メッセージのルーティングは，メッセージ自身の持つ内容によって，Publisher から Subscriber に対して行われる．このフレームワークは，既にハードウェアレベルで実装されている (e.g., Solace の XML Messaging Router[2)])．

CBN 自体は，Publisher/Subscriber 間で透過的なメッセージ配送を行うフレームワークである．すなわち，Publisher が送信したメッセージは，そのまま Subscriber に配送される．本研究では，投入されたメッセージは，ネットワーク内部で処理され，その結果がユーザに配達されるネットワークを，CBN の上に実装する状況を想定している．すなわち，メッセージの送信者は，処理の元になるデータを発行し，受信者は処理の結果を利用するユーザとして振舞う．我々の現調査段階においては，このような運用形態における研究は，まだ見つかっていない．

データ処理コンポーネントを，ネットワーク内で移動可能にし，通信遅延，処理速度，トラフィックなどを改善する一般的な手法として，Code Migration(もしくは Mobile Agent)[10)] 技術の応用が知られている．スクリプト言語や，PIAX Agent[1)]，あるいは Xen[3)] などによる仮想マシンは，すべてこの技術を持っている．ただし，データ処理コンポーネントを，ネットワーク内で移動可能になったときに，どのように移動すればよいかは，これらの技術

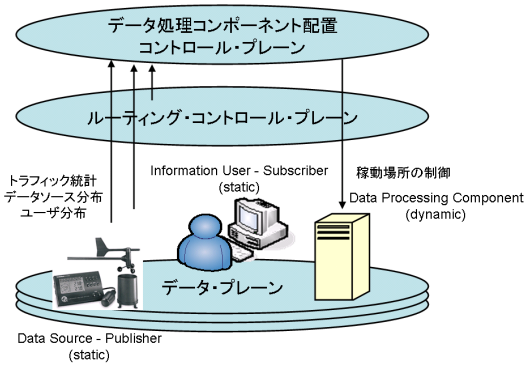

図 2 データ処理コンポーネント配置最適化のためのシステム・アーキテクチャ
Fig. 2 The system architecture for optimizing the location of data processing components

自体は対象としていない．最適配置問題は，対象とするネットワーク環境によって変わるものであり，例えば，IP ネットワーク，各種 DHT[8),9)] は，別々の計算方法を必要とする．本研究では，CBN，具体的には[11)] で述べられているようなネットワーク環境における最適配置問題のモデル化を行っている．

3. データ処理コンポーネントの配置最適化モデル

3.1 アーキテクチャ

図 2 に，システム・アーキテクチャを示す．CBN は，本来の機能として，Publisher から Subscriber までのメッセージ配送を管理するコントロール・プレーンを持っている．本研究では，このコントロール・プレーンとは独立にデータ処理コンポーネントの配置を管理するためのコントロール・プレーンを設け，既存のコントロール・プレーンに対しては，今まで通り，Publisher/Subscriber 間のメッセージ配送のみを扱かわさせるものとする．

データ・プレーンは，運用上の管理手法によって，複数枚で構成されている．例えば，

- データ・プレーン α は，気象センサの生データ
- データ・プレーン β は，ビル管理情報の生データ
- データ・プレーン γ は，地域の統計的な気象データ
- データ・プレーン δ は，警告情報の配信用

などと定義されているものとする．

Publisher(データソース) や Subscriber(ユーザ) の分布，トラフィックの統計情報は，データ処理コンポーネント配置のためのコントロール・プレーンに集められる．これらの情報から，データ処理コンポーネントの最適な配置場所を何らかのコスト基準のもと計算し，動的に配置および実行させる．

3.2 システムモデル

配置最適化を実現するためには，データソース (Publisher)→ データ処理コンポーネント → ユーザ (Subscriber) のデータフローをモデル化し，その上を流れるトラフィックの統計値を考慮することが必須である．

このデータフローモデル化において，ネットワークが複数の独立したデータプレーンで構成されていること，Content-Based Publish/Subscribe 方式であることが考慮されなければならない．そこで，以下では，Content-Based Publish/Subscribe 方式を考える上での基礎にあたるメッセージクラスについて定義を行い，それぞれのモデル化を行っていく．

こうして，最適化を行うための基礎を整えた上で，配置最適化のためのフレームワークを，3.5 節にて規定する．

3.2.1 メッセージクラス

メッセージクラスとは，メッセージを，内容的な基準において対象とする範囲を明示化するためのものである．メッセージが XML で表現されている場合，メッセージクラスは XPath で表現される．例えば，

```
/sensor[type="Temperature"]                                    (1)
```

は一つのメッセージクラスであり，このメッセージクラスに属する XML メッセージとしては，

```
<sensor id="Alice" type="Temperature">
  <value>23.5</value>                                          (2)
</sensor>
```

や

```
<sensor id="Bob" type="Temperature">
  <value>4.5</value>                                           (3)
</sensor>
```

などがある．なお，

```
<sensor id="Mary" type="Humidity" />                           (4)
```

は，上記のクラスには含まれない (理由 type=”Temperature”でないため)．

すべてのメッセージを表すメッセージクラスは，XPath 表現では，

```
/*                                                    (5)
```

となる．我々の研究では，これを M と表現する．

一般にメッセージ (m と表記する) は M の要素であり，本論文では，$m \in M$ の関係にあると表記する．またメッセージクラスは，M の部分集合として考えられる．そのため，いま3つのメッセージクラス A, B, C がある場合，$A, B, C \subseteq M$ と記述する．

メッセージクラスの間で包含関係が成立する場合がある．例えば，メッセージクラス A, B を次のように定義すると，A と B の間に包含関係が成立する．

$A =$ `/sensor[type="Temperature"]` (6)

$B =$ `/sensor[type="Temperature" and model="WXT510"]` (7)

このとき，

$$\forall m \in M(m \in B \rightarrow m \in A) \quad (8)$$

であるため，

$$B \subseteq A \quad (9)$$

である．

包含関係は，後述のデータ・プレーンに対する Publish/Subscribe を考える際に必要な考え方である．

3.2.2 Content-Based Publish/Subscribe

CBN においては，Subscriber は，購読したいメッセージをメッセージクラス形式で，予めネットワークに登録しておく．例えば，温度データを受信したいユーザ (Subscriber) は，/sensor[type=”Temperature”] を，ネットワークに登録しておくことで，それに属するメッセージを受信することができる．一方 Publisher は，暗黙のうちに送信するメッセージクラスを持っている．例えば，気圧データを送信するセンサ (Publisher) は，/sensor[type=”Pressure”] の範囲内のメッセージをネットワークに対して発信する．

P を Publisher 全体の集合，S を Subscriber 全体の集合としよう．特に，$P = \{p_1, \ldots, p_n\}$, とし，$p(\in P)$, $s(\in S)$ について考える．$F(p)$ を Publisher p が公開するメッセージクラス，$G(s)$ を Subscriber s が登録する購読条件とすると，$F(p) \cap G(s)$ は，p により発信されたメッセージのうち，$G(s)$ で受信されるものを表す．最終的に，s がすべての Publisher から受信するメッセージを $H(s)$ と表記することにすると，

$$H(s) = (F(p_1) \cap G(s)) \cup \ldots \cup (F(p_n) \cap G(s)) \quad (10)$$

もしくは，

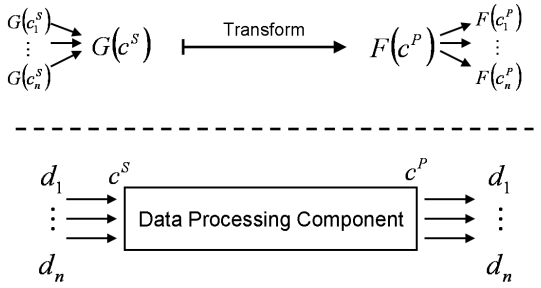

図 3 データ処理コンポーネント周辺のデータフロー
Fig. 3 Data flow around a data processing component

$$H(s) = \{\bigcup_{p \in P} F(p)\} \cap G(s) \quad (11)$$

と分解できる．これは，Content-Based Publish/Subscribe の定式化されたものであり，個別の具体的な最適化関数の設計を行う際に参照されるべきものである．

3.2.3 データ・プレーン

実際の運用では，複数枚の独立なデータ・プレーンを扱うことになる．これらデータ・プレーンの集合を D とすると，$d \in D$ は特定のデータ・プレーンを表し，これはメッセージの扱う範囲 (i.e., メッセージクラス) を持っている．以下では，このメッセージクラスを，$K(d)$ と表記することにする．

それぞれのデータ・プレーンを独立なものとして扱うように管理してある方が，経験上運用しやすいシステムとなる．これを正式な形式で記述すれば，

$$\forall d, e \in D\{d \neq e \rightarrow K(d) \cap K(e) = \Phi\} \quad (12)$$

である (ここで Φ は空集合を表す)．本研究でのデータ・プレーンは，運用管理の仕組みにより，このように分解されているものとする．

運用管理の仕組みにより与えられる制約条件がもう一つある．それは，Publisher の公開メッセージクラスおよび Subscriber の購読メッセージクラスは，全データ・プレーンの和集合に内包されていること，である．これは，それぞれのメッセージはいずれかのデータ・プレーンに属していなければいけないという制約と等しい．正式には，

$$\forall p \in P, F(p) \subseteq \bigcup_{d \in D} K(d) \quad (13)$$

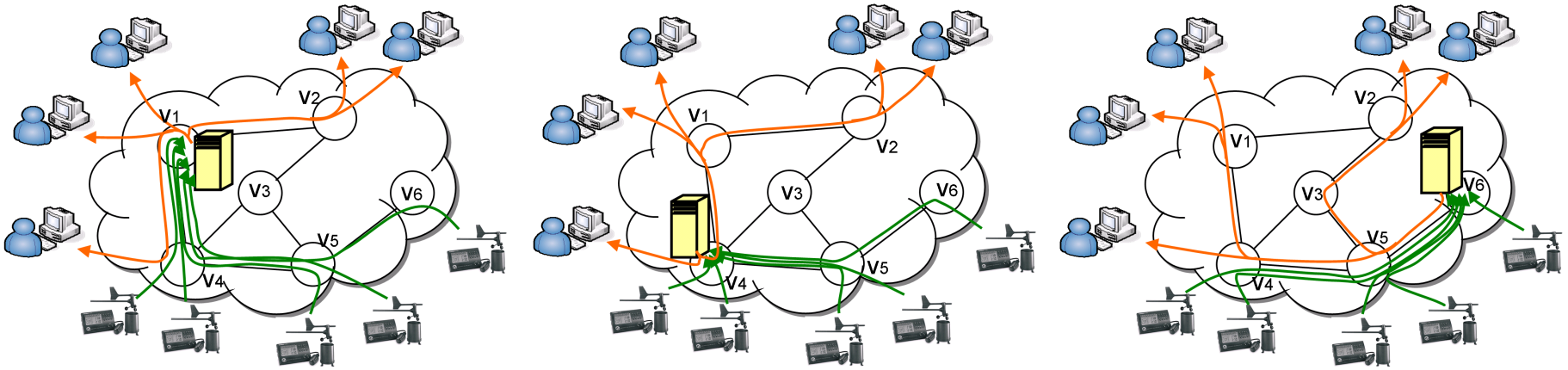

図 4 データ処理コンポーネントの配置場所とデータ配信トポロジ (データフロー)
Fig. 4 The topology of data flow and the location of a data processing component

$$\left(\Leftrightarrow \forall p \in P, \forall m \in F(p), \exists d \in D, (m \in K(d))\right) \tag{14}$$

かつ

$$\forall s \in S, G(s) \subseteq \bigcup_{d \in D} K(d) \tag{15}$$

$$\left(\Leftrightarrow \forall s \in S, \forall m \in G(s), \exists d \in D, (m \in K(d))\right) \tag{16}$$

と規定できる.

3.2.4 データ処理コンポーネント

データ処理コンポーネントは，メッセージを受信すると共に，処理結果などを発信する．一般に，データ処理コンポーネントは，Subscriber と Publisher の両方の機能を持ち，内部でデータ処理を行う (図 3)．いまネットワークに存在するデータ処理コンポーネントの集合を C とする．あるデータ処理コンポーネント $c \in C$ に対し，c^P および c^S で，c の Publisher と Subscriber を表すことにする．このときに，$F(c^P)$ は，Publisher のメッセージクラス，$G(c^S)$ は Subscriber のメッセージクラスを表すことになる．

$F(c^P)$ や $G(c^S)$ は，データプレーン $d(\in D)$ ごとに分解することが可能である (証明は省略)．c^P が d に対して発信するメッセージクラスを $F(c_d^P)$ とし，c^S がデータ・プレーン d から購読するメッセージクラスを $G(c_d^S)$ とすると，

$$F(c_d^P) = F(c^P) \cap K(d) \tag{17}$$

$$G(c_d^S) = G(c^S) \cap K(d) \tag{18}$$

で与えることができる．このようなモデル化により，データソース，データ処理コンポーネント，ユーザ間を結ぶ基礎ができあがった．

3.2.5 配置の最適化

$G = (V, E)$ を CBN のネットワーク・トポロジとする．データ処理コンポーネント c の配置場所によって，データ配信トポロジが変わる．図 4 の例は，v_1, v_4, v_6 それぞれにデータ処理コンポーネントを配置した場合の，データ配信トポロジと，データソースからユーザまでの最大ホップ数を記載している．この図の例では，最大ホップ数が最も小さくなるのは，v_1 もしくは v_4 にデータ処理コンポーネントを配置したときであることが読み取れる．このように，最大ホップ数のようなコスト基準を定め，そのコストが最小となるところにデータ処理コンポーネントを配置することが，本研究の提案する最適化方法の基本的な考え方である．なお，コスト基準の例としては，最大遅延，平均遅延，トラフィック総量などが考えられる．

ここでは，まず，データソースからユーザまでの間に，データ処理コンポーネントが 1 個しか存在しない場合について考え，その後，N 個存在する場合への一般化を行う．

データソースからユーザまでの間に，データ処理コンポーネントが 1 個しか存在しない場合．そのデータ処理コンポーネントを c，c を $v \in V$ に設置した場合のコスト関数を $\Psi(v)$ とおく．コスト基準 Ψ における c の最適な配置場所 $L_\Psi(c)$ は，次の関係を満たす．

$$L_\Psi(c) = argmin_{v \in V}\{\Psi(v)\} \quad (19)$$

本論文では，詳細は割愛するが，通常は，それぞれのコスト計算の中には，ネットワーク・トポロジや，収集された統計情報 (e.g., リンクのトラフィック量，メッセージクラスごとのトラフィック分布) が含まれる．

次に，データソースからユーザまでの間に，データ処理コンポーネントが，n 個存在する場合について考える．それぞれのデータ処理コンポーネントを $c_1, \ldots, c_n$ と置き，それぞれの取りうる配置場所を $v_1, \ldots, v_n$ と置く ($\forall i, v_i \in V$ である)．このとき，コスト関数は $\Gamma(v_1, \ldots, v_n)$ のような形態を取り，Γ を最小にする $v_1, \ldots, v_n$ の列が，$c_1, \ldots, c_n$ の最適な配置場所となる．

4. 想定される実装形態

現在の研究段階では，基本的な枠組みについて提案しているだけで，具体的な実装を行ってはいない．実装の形態は，各種考えられ，これらは，今後，議論の末に，最適なものを選び実験していく予定である．

4.1 コントロール・プレーンの実装

完全分散型，集中制御型，ハイブリッド型が考えられる．

完全分散型

完全分散型は，コントロール・プレーンでの処理を，分散アルゴリズムで実装する方式である．例えば，次のような方式で分散的に実装することができる．

(1) ネットワークに登録されたデータ処理コンポーネントの情報をネットワーク全体で共有する
(2) 各ノードはそれぞれのコンポーネントを自分の場所に置いたときのコストを計算する (コスト計算に必要な情報は，何らかの方法で既に持っているものとする)
(3) コストの計算結果を，それぞれのコンポーネントごとにネットワーク全体で共有する．
(4) 最小コストとなるノードで動かすことをネットワーク全体で同意する
(5) 選ばれたノードで動作を開始する

アルゴリズムをうまく設計することで，自律性を持たせることができれば，ネットワーク分断などによる障害にもある程度柔軟に対応できるようになるが，ノード間の相互作用が大きくなりがちなので，ネットワークの遅延が一連の処理の速度的なボトルネックとなってしまうという弱点がある．

集中制御型

すべての情報を，単一コンピュータに集めて，その場で集中的に処理を行う．この方式を使えば，コントロール・プレーンでの処理は，完全分散型と比べてかなり高速だが，コントロール・サーバが停止してしまうと，機能しなくなってしまうという欠点がある．そのため，現実的には，次のようなハイブリッド型での実装が望ましいと筆者らは考える．

ハイブリッド型

自律的に代表ノードを決定し，そこに集中制御型でコントロール・プレーンを実装する方式．第 2，第 3 の代表ノードも，同時に用意しておき，コントロール・プレーンで扱うデータをこれらの間でも共有しておけば，第 1 の代表ノードがアクセス不能になった場合に，短時間で代表ノードの切り替えを行うことができる．比較的，耐故障性がよく，コントロール・プレーン処理も高速に行うことができると思われる．

4.2 データ・プレーンの実装

ここでは，データ・プレーンの分類管理方法と，データ処理コンポーネントの実装形態について整理する．

4.2.1 データ・プレーンの分類管理

CBN の中には，複数のデータ・プレーンを作成し，気象観測データ，大域統計気象データ，警戒情報，などの各種データや情報の配信を，別々に考えるべきであると述べた．ここでは，XML-based CBN において，ネットワークオペレータが，どのようにそれらを管理するかについて述べる．

ネットワークオペレータは，データ・プレーンを管理するための表 (スキーマ) を持っている．その表は，タグ名による分類によるものと，名前空間による分類によるものがあるだろう．具体的には，次のようになっているものと思われる．

<<タグ名による分類>>

■データ・プレーン 1

メッセージクラス: /sensor
データ種別: 気象観測データ
具体例: `<sensor ... >...</sensor>`であるすべての XML メッセージ

■データ・プレーン 2

メッセージクラス: /aggData

データ種別: 大域統計気象データ

具体例: <aggData ... > ... </aggData>であるすべての XML メッセージ

■データ・プレーン 3

メッセージクラス: /warnInfo

データ種別: 警戒情報

具体例: <warnInfo ...>...</warnInfo>であるすべての XML メッセージ

<<名前空間による分類>>

既存のものを統合しようとした場合や，データフォーマットのバージョンを管理するために，XML では名前空間を規定することができる．そこで名前空間を使って，データ・プレーンを分類することも可能である．

■データ・プレーン 1

XML 名前空間: http://live-e.org/DataType/2007/03/

データ種別: Live E!データフォーマット規定 (Ver.200703)

具体例: xmlns=”http://live-e.org/DataType/2007/03”に属するすべての XML メッセージ

■データ・プレーン 2

XML 名前空間: http://live-e.org/DataType/2009/07/

データ種別: Live E! データフォーマット規定 (Ver.200907)

具体例: xmlns=”http://live-e.org/DataType/2009/07/”に属するすべての XML メッセージ

4.2.2 データ処理コンポーネント

データ処理コンポーネントを，各 CBN ルータの近傍に設置されたプラットフォーム上で実行させることを考える (図 5)．プラットフォームは，データ処理コンポーネントを動的に結合できるように設計されており，データ処理コンポーネントは，どのプラットフォームに接続されても，内部に何ら変更を加える必要なく，同じサービス内容を提供することができるようにする．このようにすることで，データ処理コンポーネントは，それぞれのプラットフォームで実行可能な Mobile Agent として実装することが可能になる．

上記のような基本的な枠組みを実現するにおいて，Mobile Agent を (1)Script で記述するか，(2)Java コードで記述するか，(3)Xen のような仮想マシンイメージで記述するかなどの選択肢が考えられる．

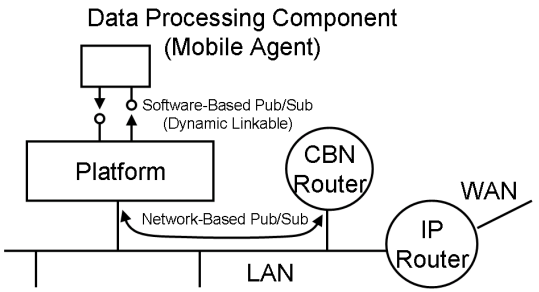

図 5 データ処理コンポーネントの実装例
Fig. 5 The platform for data processing component deployment

5. おわりに

本論文では，Content-Based Network(CBN) にデータ処理機能を持たせるシナリオを取り上げ，データ処理コンポーネントのネットワーク内での最適配置問題について扱った．最適配置の基準としては，最大遅延，平均遅延，トラフィック量などが考えられるが，本論文では，具体的な最適化を実現する前段階として，データ処理を含む CBN のモデル化の試みについて述べた．データ処理コンポーネントの配置最適化を実現するためには，データソース，データ処理コンポーネント，ユーザの間のデータフローをモデル化する必要があった．そのため，CBN の基礎にあたるメッセージクラスを取り上げ，Content-Based Publish/Subscribe のモデル化，そしてデータプレーンの管理についてのモデル化について言及した．

本論文では，コントロール・プレーンやデータ・プレーンの実装方式についても提示した．コントロール・プレーンは，完全分散型，集中管理型，ハイブリッド型での実装が考えられる．またデータ・プレーンの分類管理においては，タグ名による分類や，XML 名前空間による分類などが考えられる．データ処理コンポーネントを Mobile Agent として実装するために，スクリプト方式，Java 方式，Xen 方式がある．

具体的な最適化関数の設計，実装による動作確認およびパフォーマンス測定などが今後の研究課題として残されている．

Vol.2009-DPS-139 No.12
2009/6/19

参 考 文 献

1) : PIAX. http://www.piax.org/.
2) : Solace Systems. http://www.solacesystems.com/.
3) : Xen. http://www.xen.org/.
4) Carzaniga, A., Rutherford, M.J. and Wolf, E.L.: A Routing Scheme for Content-Based Networking, *In Proceedings of IEEE INFOCOM 2004* (2004).
5) Carzaniga, A., Wolf, A. L. and Wolf, E. L.: Content-Based Networking: A New Communication Infrastructure, *LNCS*, Vol.2538, pp.59–68 (2001).
6) Eugster, P., Felber, P.A., Guerraoui, R. and Kermarrec, A.-M.: The Many Faces of Publish/Subscribe, *ACM Computing Surveys*, Vol.35, No.2, pp.114–131 (2003).
7) Ochiai, H., Wang, Z., Oguchi, R., Sugiyama, A., Sakamoto, Y., Ishida, S. and Esaki, H.: Application of Content-Based Network for Sensor Data Distribution System, *In Proceedings of IEEE/IPSJ SAINT* (2007).
8) Ratnasamy, S., Francis, P., Shenker, S., Karp, R. and Handley, M.: A Scalable Content-Addressable Network, *In Proceedings of ACM SIGCOMM*, pp. 161–172 (2001).
9) Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F. and Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications, *In Proceedings of ACM SIGCOMM*, pp.149–160 (2001).
10) Tanenbaum, A.S. and Steen, M.V.: *Distributed Systems*, chapterCode Migration, pp.103–112, Peason Education, Inc., second edition (2006).
11) University, A.C., Carzaniga, A. and Wolf, A.L.: Forwarding in a Content-Based Network, *In Proceedings of ACM SIGCOMM*, pp.163–174 (2003).