追記・参照型データ管理システムの設計と評価

赤間浩樹[†1] 内山寛之[†1] 西岡秀一[†1,*1]
内藤一兵衛[†1] 谷口展郎[†1] 長谷川知洋[†1]
兵藤正樹[†1] 三浦史光[†1]
山室雅司[†1] 櫻井紀彦[†1]

センシング機器（センサ）や IC タグなどから発生する多量の情報，携帯機器の発達により収集される様々な個人の活動履歴情報，ブログのようにコンシューマが生成する多量の情報など，ユビキタス社会の進展にともなって時々刻々と多種多様な情報が生まれ，ネットワークに流れ込んでいる．これら多種多様で追記型（Write Once）の情報に対して，適切に蓄積・管理するとともに，利用者の目的に合った形で情報を統合し，検索や分析といった参照（Read Many）を可能にするプラットフォームが必要になる．本論文では，このプラットフォームへの要件を定義し，それを実現するアーキテクチャとして，データの流れ方向と処理量の増加方向に対する 2 つの分散軸を持ち，系としての連続性を維持したまま機能的成長・規模的成長を実現する追記・参照型 DMS アーキテクチャを提案する．そのプロトタイプシステムを設計・構成し，映像監視システムへの適用を通して成長への対応力をスケーラビリティと系の連続性の面から評価する．

Design and Evaluation of a Data Management System for WORM Data Processing

Hiroki Akama,[†1] Hiroyuki Uchiyama,[†1] Shuichi Nishioka,[†1,*1]
Ichibe Naito,[†1] Noburou Taniguchi,[†1] Tomohiro Hasegawa,[†1]
Masaki Hyodo,[†1] Fumiaki Miura,[†1] Masashi Yamamuro[†1]
and Norihiko Sakurai[†1]

Vast and various data streams are being generated from miscellaneous data sources such as RFID sensors, web logs, cellular phones logs and web cams. To deal with these data streams, we propose an architecture that receives data streams, executes diverse operations defined by users and stores/alerts the results. This architecture is composed of PC servers and has three main features: (1) scalability for operation processing, (2) dynamic updates of the operations, (3) automatic deployment and execution of operations. We describe a prototype system based on the architecture and a video monitoring application implemented on the system. Finally, we evaluate both system scalability (in terms of the numbers of PC servers) and the continuity of stream processing while some operation is being updated dynamically.

1. はじめに

今後のネットワーク社会では，広域に分散する，センサ情報，情報処理システムやネットワークのログ情報として吸い上げられた情報が，マーケティング情報，安全管理情報などとして分析・活用されていく．すでに情報源へのアクセス時の操作履歴を記録することでサービスのマーケティング分析やリコメンドを行うなどのサービスが出現しており，これらは，過去の行動履歴を管理・活用したサービスへと展開しつつある．経済的にリーズナブルな情報格納手段と検索手段が整えば，半永久保存された過去のセンシング情報や WWW などの情報源へのアクセスログ情報を新たな視点・目的で活用し，異業種間で連携した情報提供サービスや，各個人対応に生涯サポートを行うようなサービスの出現が期待できる．

このような社会背景から，情報爆発時代に向けた新しい IT 基盤の研究[1] プロジェクト活動や，各種セン

†1 日本電信電話株式会社 NTT サイバースペース研究所
NTT Cyber Space Laboratories, NTT Corporation
*1 現在，NTT レゾナント株式会社
Presently with NTT Resonant Corporation

サ向けプラットフォーム（PF）の研究開発，ネットワーク・フォレンジクス・ツールなどのログ記録システムの開発などが活発化している．

我々はネットワークに大量発生するログ情報を，収集・構造化・半永久保存し，複数の情報を同期・相互連携し，企業や個人の各々の目的・条件に応じたフィルタリングや，時間・空間をまたがった検索などを可能にする情報統合管理サービスを目指して，追記・参照型データ管理システム（以下 DMS: Data Management System）の研究開発を進めている[2),3)]．

1.1 情報統合管理サービス PF の基本コンセプト

我々の目指すサービスを実現するための PF イメージを**図 1** に示す．図の下側には情報源となる各種のセンサ群があり，図の上側には情報を活用する各種応用サービスが存在する．従来は，情報源と応用サービスは密に結び付き，個々に構築・運用されていたが，社会的インフラともいえる情報源の増加にともなって，それら情報源を複数の応用に活用するというニーズが高まるのは必然である．そこで，我々の DMS は，それら情報源と応用サービスの間に存在し，時間や空間を越えて情報を仲介接続し，新たな価値を創出することを目指しており，以下の 2 点をコンセプトとして持つことが特徴である．

【特徴 I】成長に対する系の連続性（新陳代謝）

情報統合管理サービス PF の情報源の 1 つであるセンサ情報系においても，全体としては，ネットワークワイドに連合した巨大な系となる．一方で，個々の要素であるセンサ機能は，逐次古いものを捨てて，新しい機能を持つセンサに更改され，それ自身はシステム寿命に関係なく技術革新と経済原則に基づいて世代交代を繰り返していく．このとき，全体の系としての連続性を維持したまま，世代の異なるセンサの共存や世代交代を受け入れていくことが要求される．さらにサービス自身の世代交代に関しても新旧世代の共存を系全体としては許容することが必要である．そのためには，新陳代謝を行うように，動的かつ部分的に世代交代を繰り返すことにより，システム全体としての連続性を維持したまま機能面と規模面の両面で成長することが要求される．

【特徴 II】半永久保存性

半永久保存のためには，情報の蓄積はできるだけ情報量の損なわれていない生データの形で保存することが望ましい．なぜならば，不用意な変換による情報量の喪失を可能な限り防ぐべきであり，基本的には，半永久保存の間，利用された世代の入出力方法とともに蓄積管理することにより，将来の新しい入出力方式に

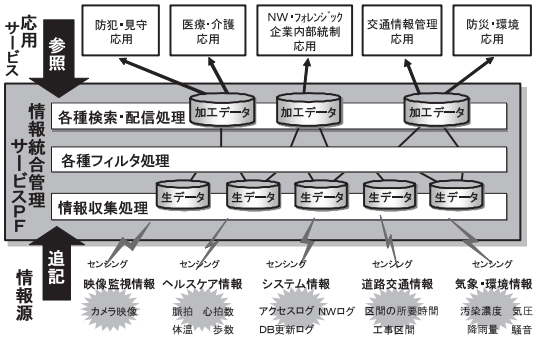


図 1 情報統合管理サービス PF
Fig. 1 Information integration service platform.

よる過去の情報へのアクセスを可能とする機能が必要である．ストレージの低価格化も驚くべきスピードで進んでおり，これまで蓄積できなかった生データをすべて蓄積することも不可能ではなくなりつつあるが，過去の生データを半永久保存することはいまだ現実的ではなく，過去にまたがってのログ情報に効率的にアクセス可能な格納形式を明確にする必要がある．ただし，半永久保存性に関しては，今後の課題として本論文では扱わない．

1.2 DMS への要件

多数の応用に長期的に対応し続けるために DMS に求められる要件は，

(1) 情報源（連続するデータ）
(2) 応用サービス
(3) データへの処理内容

の各々の対象に対応して，多様化，追加，変更（含，削除）といった**機能的成長**と**規模的成長**を PF としてサポートすることである．より具体的には，

- 機能的成長による新たな機能と陳腐化した機能の入替えを，系として連続性を失わない中で行えること（**連続データに対する系無中断の動的機能入替え**），
- 機能的成長にともなう処理の複雑化や規模的成長によるシステムの負荷の増加に対して，マシン追加による系能力の追加と負荷への追随を，系として連続性を失わない中で行えること（**連続データに対する系無中断のスケールアウト**），

の 2 点の実現を目指す．なお，同一マシンに CPU を追加する SMP（Symmetric Multi Processing）構成の活用や CPU の置換によってスケーラビリティを向上させることをスケールアップ，PC クラスタなどマシンを追加することでスケーラビリティを向上させることをスケールアウトと呼び，本論文では区別する．

以下，本論文では，上記の要件解決に向けた DMS

アーキテクチャを 2 章で提案し，類似技術との比較と優位性の考察を行う．3 章で，本アーキテクチャに基づいて設計し実装したプロトタイプを示し，4 章で我々のアーキテクチャが 1 章の各要件にどう対応し実現するかを述べる．5 章で応用システムの構築による本アーキテクチャの応用適用性の確認と，スケールアウト能力や系無中断の動的機能入替えに関する評価実験による成長への適応能力を示し，6 章で今後の課題を考察する．

2. 追記・参照型 DMS アーキテクチャ

2.1 アーキテクチャの特徴と効果

連続するデータへの系無中断の動的機能入替えと，系無中断のスケールアウトの両立を実現するため，DMS は以下のアーキテクチャをとる．なお，現状のアーキテクチャは 5 章の応用で述べるような不正 SQL 検出のための TCP/IP パケット検査や Web カメラの映像監視などのようなストリームデータの処理を対象とし，全国から集めた情報をセンタに設置された数百台の PC クラスタで処理することを想定している．

2.1.1 系の機能分割，配置の自由化

Web カメラなどから流れ込むストリームデータに対する処理を，データの受付を行う**追記部（Q）**，受付データに対し多様な処理を行う**フィルタ部（F）**，処理後のデータを各種応用サービスに提供する**ビュー部（V）**の 3 層に分割する．さらに分割された各機能は系内のマシンに自由に配置可能にする．この分割は，ストリームデータに対する高度で負荷の高い処理がフィルタ部（F）で行われるという世界観の中で，データの受付とキューイングに専念する追記部（Q）とデータの参照を提供するビュー部（V）をフィルタ部（F）から分離するという考え方による．なお，この分割にともなって発生する機能間の接続については後に述べる．

2.1.2 機能の並列構成，フィルタ部（F）完全性，ビュー部（V）での合流機能

分割された各機能は数百台からなる PC クラスタ上で処理される．フィルタ部（F）は系内に複数（DMS 管理者が設計したプロセス数分）存在し，各々が並列に動作する．フィルタ部（F）は任意の情報源からのデータに対して処理が可能であるという特性（フィルタ部（F）の完全性）を持つ．そのため系内の全フィルタ部（F）は等質となる．この完全性を使って，1 つの追記部（Q）で受け付けたストリームデータの各レコードを複数のフィルタ部（F）で並列に処理することになるが，その分散処理された結果の合流はビュー部（V）によって行われ，さらに系内に流れるデータに自動付与される時刻印によってデータは整列される．

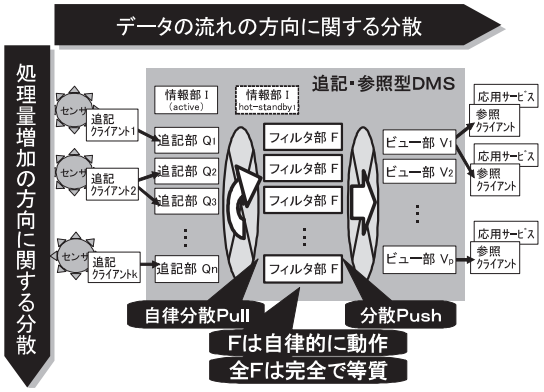


図 2 DMS アーキテクチャ
Fig. 2 Architecture of the DMS.

一方，追記部（Q）は情報源側の接続要求によって生成または割り当てられ，並列に動作し，接続が開放されるまでは特定の情報源からのストリームデータを専門に受け付ける．情報源側が複数の追記部（Q）にラウンドロビンなどで並列に追記を行った場合のデータの合流と整列もビュー部（V）によって行う．

このように，DMS は横と縦の 2 軸の分散構成を持ち，それぞれが系内の自由なマシンに配置できる構成（**図 2**）となる．この 2 軸の分散の前者を**データの流れの方向に関する分散**，後者を**処理量増加の方向に関する分散**と呼ぶ．なお，処理量の増加には情報源の増加によるものと，各データへの処理の複雑化によるものの両方が含まれる．DMS のアーキテクチャはこの 2 軸の分散に加え，さらに以下の特徴を持つ．

2.1.3 フィルタ部（F）の自律性

DMS における各種処理制御や判断の主体はフィルタ部（F）である．これをフィルタ部（F）の自律性と呼ぶ．多数の追記部（Q）で受け付けたデータについて，各フィルタ部（F）が主導権を持って追記部（Q）の決定を行い，決定した追記部（Q）からのデータの取得（Pull）処理を行う．同様にフィルタ部（F）で処理した結果をビュー部（V）に送る処理もフィルタ部（F）が主導権を持つデータ送信（Push）処理となる．このフィルタ部（F）が自律性を持つモデルは，例えれば完全で主体的な従業員が多数の仕事を処理するモデルと考えることができる．このモデルでは人の追加が容易に可能であり，また系全体としての能力さえ維持できれば人が何人休んでも系としてサービスの連続性が維持できる．DMS はこの特性を利用している．なお，各情報源のストリームデータに対して，どの処理群を実行し，どのビュー部（V）に送るのかといった情報や，系内のプロセス情報の共有，追記部（Q）

の状態などは次章で詳述するマルチキャストによる緩やかな情報同期などを活用して事前に各部に共有されているものとする．

さて，フィルタ部（F）の主導権を実現するため，追記部（Q）とフィルタ部（F）間のデータ配信制御方式は，追記部（Q）がフィルタ部（F）に Push するのではなく，フィルタ部（F）が追記部（Q）からデータを Pull する方式になっているが，これにより以下の効果が期待できる．

効果 1. フィルタ部（F）が過負荷になることを防ぎ，系としての安定性が向上する．

効果 2. フィルタ部（F）が自らのタイミングで仕事の制御（処理の実行，停止，再起動など）が可能となる．

逆に追記部（Q）がフィルタ部（F）にデータを Push する方式にした場合に考えられる各種方式は以下の理由により採用しなかった．

- フィルタ部（F）が受付データを一時的に蓄積する方式は，追記部（Q）との機能重複が発生し機能分割の目的に反する．
- フィルタ部（F）の過負荷を避けるためにフィルタ部（F）が追記部（Q）の Push を断る方式は，追記部（Q）に処理先のフィルタ（F）を見つける処理コストが付加されるが，追記部（Q）の処理負担が増加する方式は本アーキテクチャの世界観（処理負荷はフィルタ部（F）側に負担させるべき）に反する．
- フィルタ部（F）が過負荷となる場合にデータを捨てる方式は，センサ種別や応用サービスによっては有効な場合もあるが，本アーキテクチャではデータを捨てずに対処可能な方式を標準とする．

以上，述べたように DMS は自律的なフィルタ部（F）を中心にして，分割された機能間と並列する機能間を接続するアーキテクチャとなっている．このアーキテクチャの効果は，まずストリームデータの処理中に，フィルタ部（F）が自律的に系参加可能（動的フィルタ部（F）の系参加）となる点にある．これにより，本アーキテクチャの狙いの 1 つである系無中断のフィルタ部（F）のスケールアウトが可能になる．さらに，もう 1 つの効果は，ストリームデータの処理中に，任意のフィルタ部（F）の判断するタイミングでそのフィルタ部（F）は系から離脱（局所的フィルタ部（F）の停止）が可能となる点にある．これによって，十分な系の規模（マシンの台数）があって，各機能が適切に分散していれば，局所的なフィルタ部（F）の停止，機能入替え，動的フィルタ（F）の系参加，を系全体のフィルタ部（F）に対して繰り返すことで，連続するデータへの系無中断の動的機能入替え（機能の追加，更新，削除）が可能になる．

2.2 類似技術との比較と考察

DMS の狙う機能成長と規模成長について，周辺分野技術との関係を整理し，DMS の優位性を考察する．

(1) 負荷分散（LB，Load Balancing）との比較

現在，多くの Web サービスは，スケールアウト構成を採用し，数百台からのマシン規模で運用されることも珍しくはない．その中心となる技術は http サーバ群のフロントに設置する LB[4] とキャッシュ技術であり LB を使って機能成長と規模成長をおおむね両立させているように見える．しかし，LB の設置が DB サーバのスケールアウトに対して本質的な解決になっていない（極端な例でいえば，特定のレコードへの更新負荷を LB することはできない）ことからも分かるように，LB の設置によって実現している機能成長と規模成長はあくまで非連続データへの参照を中心としたものであって，追記を含む更新系や映像ストリームなどの連続データには十分対応できない．一方，DMS は，連続データの追記に対して系の規模成長と機能成長を実現可能である．

(2) 情報源ごとに独立した PC を割り当てる構成との比較

各情報源が独立で分離されているのであれば，情報源ごとにマシンを割り当てるという方式をとるという考え方もある．しかし，各情報源の能力が増加（たとえば Web カメラが高精細化）した際に必要となるマシンの増強や，各情報源からのデータへの処理が増加（たとえば Web カメラへの処理を顔検出から顔認識に処理変更）した際に必要となるマシンの増強や，サービスの高可用化や無停止機能入替えを狙って Active-Standby 構成とした場合に系の半分が Standby となり系の分割損が大きくなる点や，複数の情報源に対する処理結果の流通制御が難しい点など，大規模なプラットフォームのアーキテクチャとしては多数の問題がある．一方 DMS は，連続するデータへの系の機能成長と規模成長を可能にしていることで情報源の能力増加や各情報源への処理増加に対しても無停止でマシン追加による系の能力追随が可能になっている．また，追記部（Q）とフィルタ部（F）は全 Active 構成であり Standby マシンが存在しないことで，系としての分割損も少ない．

(3) 分散データストリーム管理システム（DSMS）との比較

DMS は既存のデータストリーム管理システム

（DSMS）を参考にして，その大規模化を目指して設計を行っており，技術分野としては分散 DSMS が最も近い．

既存の DSMS は，ストリームデータを時刻印が付いた無限のタプル列と見なし，事前に指定した問合せを繰り返し適用し続ける連続問合せ（Continuous Query）という仕組みや，ストリームから有限のリレーションを抽出し処理する window という仕組みを持っている．我々の DMS も後述するような SQL フィルタなどで Continuous Query を提供し，選択や射影を指定することができ，また，window 処理も可能にしている点で，基本的には DSMS である．そして，我々の DMS は以下に述べるいくつかの点で従来の DSMS に対して優位性を持つ．

まず，負荷への対応の面で比較を行うと，従来の代表的な DSMS である STREAM[5] や Aurora[6] は，SMP によるスケールアップも可能であるが，ストリームデータの到着レートが過剰でストリームの処理が困難であると判定した場合には，到着データに対してユーザが定義する QoS に基づき自動的にフィルタリングやサンプリングを行い処理対象のデータ量を減らす load shedding 機能によって対処を行う．また，Gigascope[7] は，低レベルと高レベルの 2 層のアーキテクチャを持ち，リングバッファを使った高速な TCP/IP パケットの集約処理を低レベルで行い，高度な処理はデータ量を削減してから行うことで高トラフィックに適用することも可能にしている．これに対し，DMS は，追記クライアント（AC）が複数の追記部（Q）に接続して分散書き込みすること，分散フィルタ処理することで，過剰なレートのデータ到着であってもデータを捨てず系の能力を増強（スケールアウト）して対応する．また，低レベルの処理がうまく設定できないような処理にも適用が可能である．

従来の DSMS を分散構成に発展させたものもある．ただしこの場合の分散の狙いは主にストリームデータに対するオペレーション群の分割と配置の最適化であり，たとえば，Aurora*[8] や STREAM（商用版は Coral8）の分散対応では分割したオペレーションを異なるサーバに配置し，連続するストリームに対してパイプライン処理を行うことを可能にする．また，Gigascope の分散対応でも複数のサーバを起動してソース IP でのハッシュにより分散することができる．しかし，たとえば顔認識のような負荷の高い処理が 1 つ存在すればパイプライン処理では系としてのレスポンス悪化の連鎖を解消できず系が破綻するし，特定の Web カメラのようにソース IP が 1 つのものについてはハッシュによる分割は利用できない．これに対し，DMS では 1 つのストリームを複数のサーバで同時並行して処理することで対処できる．STREAM ではストリームを分割し LB 技術を組み合わせることも主張しているが，たとえば顔認識の処理が重い場合に顔の含まれる映像を効率的に LB することは難しいし，逆にそれを厳密に行えば LB が系のボトルネックとなる．一方，DMS はフィルタ部（F）の自律性により処理が終わったものから順に次のデータを処理することで，結果的に負荷の高い顔画像が含まれたデータへの処理が分散される．また，複数の情報源からの処理が混在しても全フィルタ部（F）が関わるため LB の範囲などの設計が不要となる効果もある．

次に，可用性の面での比較を行う．従来の DSMS は各 Active サーバに対して Standby サーバを用意する Active-Standby 構成をとっているものが主流である．これはストリームデータを Active サーバと Standby サーバの両方で処理し，ハートビートによって Active サーバの故障を検出すると Standby サーバの処理結果を後段のサーバに流すものである．また，StreamSpinner[9] は仮想マシン技術を利用し，チェックポイントの契機で仮想マシン全体を別マシンにバックアップすること（Passive-Standby）で高可用性を実現している．一方，DMS は 1 つのストリームを複数の PC サーバによって並列（全 Active 構成）に処理していることで，系としての分割損が少ない状態で高可用性を実現できる．また，この全 Active 構成を活用した部分的なフィルタ部（F）の処理入替えを活用すれば，Continuous Query の追加だけでなく，OS の入替えさえも系として無停止の中で可能にする．

このような方式の違いから，従来の DSMS において連続データに対する系無中断の動的機能入替えと，連続データに対する系無中断のスケールアウトを十分に両立させているものはなかった．

(4) データベース管理システム（DBMS）との比較

データベース管理システム（DBMS）においてスケールアウトは長年の課題であり，更新キャッシュのメモリ間同期を利用した試みなどが存在するが，現状はスケールアップでの解決か，ユーザ ID やロケーション ID などを使った設計によって更新を分散管理する方法がとられており，DMS のような機能成長と規模成長の両立を実現しているとはいえない．もちろん，DMS は追記と参照に特化しており更新が行えない点やトランザクションを持たない点など DBMS を置き換えるものではない．

DMS のアーキテクチャではビュー部（V）の 1 種

類として DBMS が分散して存在する利用方法を想定（現状は PostgreSQL をビュー部（V）に配置する構成が可能）しており，DMS は分散 DBMS に対しデータ投入とトリガ機能部分を分散系に拡張した系ととらえることも可能である．従来の Harmonica[10] は，ストリーム処理エンジンと DBMS を統合し，ストリームデータに対する問合せやストリームデータを履歴データとして格納する機能を提供する．しかし，DMS のように規模成長や機能成長を解決するものではない．

3. 追記・参照型 DMS の設計と実装

2 章で述べたアーキテクチャをもとに追記・参照型 DMS を設計しプロトタイプとして実装した．DMS プロトタイプは主に C++（Java クライアント用ライブラリは Java，運用系コマンドは Perl）を使用し，IA マシン（EM64T）上の Linux（Fedora Core）で動作している．系内通信や AC との通信には標準データ形式（XDR）を使っているため，32 bit/64 bit マシンの混在環境へも対応可能である．分散構成を基本とする DMS は現在 50 台の環境での動作実績や 50 台の Web カメラの接続実績があり，1 台構成のスモールスタートからの動的なスケールアウトが可能である．本章ではその設計と実装を示す．

DMS を使ったサービス構築は，情報源側の**追記クライアント（AC）**作成者，フィルタ処理の要素機能である**フィルタ部オペレーション（FOP）**作成者，応用サービス側でビューを利用する**参照クライアント（RC）**作成者，そして DMS 管理者の 4 者の作業によって行われる．

各情報源は情報源情報（情報源 ID，情報源のデータのスキーマ定義，追記部（Q）での生データ永続化フラグ，処理 ID 群，追記部（Q）でのフィルタ部（F）処理応答待ち時間，ストリームデータの Window 幅，追記部（Q）状態通知閾値など）を持ち，情報源 ID によって識別される．DMS 管理者は，この情報源情報を管理する．たとえば，新たな情報源の追加の際には，追記クライアント（AC）作成者が DMS 管理者に情報源 ID の払い出しを要求する．情報源 ID のスキーマが変更される場合にも同様となる．さらに，DMS 管理者は 4 者間の利害調整を行ったうえでフィルタ部（F）での処理を定義した処理情報（処理 ID，フィルタ部オペレーション（FOP）の系列，処理 ID の版番号）を管理する．情報源情報内の処理 ID や処理情報内のフィルタ部オペレーション（FOP）は複数列挙することが可能で，1 つの情報源 ID から系に入力されたデータは，その情報源 ID 内の処理 ID で特定されるフィルタ部オペレーション（FOP）系列が順に適用されていく．たとえば，情報源 ID として CAM-Yokosuka-001 があり，データスキーマは Schema-CAM（image blob, city string）からなり，city カラムに “Yokosuka” が入っているデータに対して顔検出を実行し，その結果を 192.168.xx.xx:40010 にあるビュー部（V）の DB（ViewName1）に送るのであれば，フィルタ部オペレーション（FOP）系列として

SqlFilter(“SELECT image FROM Schema-CAM WHERE city=’Yokosuka’ ”);
FaceDetect();
Emitter(“192.168.xx.xx:40010:ViewName1”);

のように記載する．

情報源情報と処理情報は，後に述べる情報部（I）に格納され，自動的な遅延同期または DMS 管理者からの指示によって追記部（Q），フィルタ部（F）に同期される．以下，ストリームデータの流れに沿って追記部（Q），フィルタ部（F），ビュー部（V）を説明し，そして系全体の管理に関わる情報部（I）とマシンの管理に関わるポートマッパ（P）を説明する．

3.1 追記部（Q）

追記クライアント（AC）からの新たな接続があった場合には系内にマルチキャスト（MC）でその接続情報を共有する．追記クライアント（AC）からの接続は情報源 ID をともなって行われるため，追記部（Q）はその情報源 ID で指定されるスキーマに割り当てられる．追記クライアント（AC）のコネクションと追記部（Q）はつねに 1 対 1 に対応するが，1 つの追記クライアント（AC）が複数の追記部（Q）とのコネクションを持つことが可能である．追記部（Q）は追記クライアント（AC）から追記されたデータを受け取り，系内管理用のレコード ID を付与するとともに，受取りの時刻印を付与し，フィルタ部（F）向けにキューイングすると同時に，生データとして DISK に書き込む．DISK 書き込みデータは時刻または容量で順にファイル切替えされ，古くなったファイルはシンボリックリンクを残して別の DISK に退避する．

追記クライアント（AC）：たとえば DMS にデータストリームを追記するクライアントは DMS が提供する追記クライアントライブラリ（C++ または Java）を使い，情報部 IP アドレスと port 番号および情報源 ID の設定（prop->setProperty(···);），ドライバ作成（dr = new AddDriver(prop);），情報部に対する接続依頼（conn = dr->getConnection();），スキーマ取得（sch = dr->getSchema();），レコード作成（rec = createRecord(sch);），データ送信（res =

conn->sendRecord(rec);）の繰返し，を行う．

3.2 フィルタ部（F）

フィルタ部（F）は追記部（Q）からデータを取得する前半部分と，受け取ったデータを処理する後半部分からなる．

前半部分は**自律的分散データ取得処理**によって系内の追記部（Q）群から 1 つ追記部（Q）を決定し，その追記部（Q）上のキューからデータを情報源 ID とともに取り出す（Pull）．フィルタ部（F）はその際，自らが持つフィルタ部（F）内の処理 ID の版番号を追記部（Q）に通知し，追記部（Q）は自分の持つデータに必要とされる処理 ID の版番号と比較し，版番号チェック結果も返却する．

後半部分は受け取ったデータに対応する情報源 ID に登録された処理 ID 群，つまりその中に含まれるフィルタ部オペレーション（FOP）系列を順に適用する．フィルタ部（F）の処理が成功した場合には追記部（Q）上のデータの削除を行い，もしフィルタ部（F）の処理が失敗した場合には Timeout 後に追記部（Q）上のデータは別のフィルタ部（F）によって処理されることでデータの紛失を防止している．ただし指定回数の失敗が発生すると ERROR データとして処理される．フィルタ部（F）はフィルタ部オペレーション（FOP）の組み込みインタフェース（I/F）を持ち，ユーザ定義処理を複数組み込むことが可能である．複数のフィルタ部オペレーション（FOP）が組み込まれたフィルタ部（F）は **F 実行モジュール**として構成（make）され，3.4 節で述べる情報部に登録する．

フィルタ部（F）はデータストリーム管理システム（DSMS[5)–9)]）の特有の処理として件数ベースの window 処理（現在のデータを起点として過去のデータを window 幅で指定される件数分だけフィルタ部オペレーション（FOP）の内部で参照すること）が可能である．なお，過去のデータについては，複製して追記部（Q）からフィルタ部（F）に流すことによって，並列動作するフィルタ部（F）での window 処理を可能にしている．

フィルタ部オペレーション（FOP）：利用者が組み込むことが可能である．フィルタ部オペレーション（FOP）作成者は，フィルタ部オペレーション（FOP）のインスタンス初期化処理（sampleFop::sampleFop(Schema& in, Schema& out) : FilterOperation(in, out){···}），終了時処理（sampleFop::~sampleFop(){···}），1 レコードを受け取ったつど実施する処理（Record* sampleFop::doExecute(char *opt, Record** recs, int ws, Record *retRec){···}），を記述し，フィルタ部（F）の構成に追加して，F 実行モジュールを作成する．なお，opt は処理情報でフィルタ部オペレーション（FOP）を記述した際に指定された引数，ws は window のレコード数，recs には ws 分のレコードが格納されて doExecute() が呼び出され，retRec を経由して次のフィルタ部オペレーション（FOP）にレコードが渡される．

標準で組み込み済みの汎用的なフィルタ部オペレーション（FOP）として，簡単な SQL 文でフィルタ条件を指定可能な SqlFilter と，処理結果をビュー部（V）組み込みライブラリに送信する Emitter がある．

SqlFilter は入力データに対して選択（selection）と射影（projection）からなる連続問合せ（Continuous Query）を可能にする．選択は引数の変更によって処理変更が可能であるが，射影はデータ列構成が変わるためフィルタ部オペレーション（FOP）自身の変更と F 実行モジュールの変更によって処理変更を行う．

例） *SqlFilter(“SELECT name, phone FROM myStrm WHERE age>20”)*

Emitter は受け取ったデータを指定したビュー部（V）へ送信する．引数としてマシン IP アドレス，Port 番号，ビュー名（ビュー部（V）が DBMS の場合は DB 名）を持つ．処理情報のフィルタ部オペレーション（FOP）系列内に Emitter を 2 つ並べれば 2 つのビュー部（V）に同じデータを送ることが可能になる．配信先を動的に切り替える場合には Emitter 内にデータのカラムを見て配信先を切り替える処理コードを記述する．

例） *Emitter(“192.168.xx.xx:40020:MyViewName2”)*

自律的分散データ取得処理：データ取得処理部のアルゴリズムは交換可能な設計となっている．各フィルタ部（F）は，情報部（I）から定期的に送られてくる系内プロセス構成情報（3.4 節参照）と追記部（Q）からの接続情報によって系内の追記部（Q）のプロセス情報を把握し，組み込まれた自律的分散データ取得処理によって接続先追記部（Q）を決定し，接続し，データの取得を行う．DMS はいくつかの自律的分散データ取得処理の実装を持つが，本論文では最も単純なラウンドロビン法での評価を行っている．

3.3 ビュー部（V）

ビュー部（V）はデータを参照クライアント（RC）に提供するものである．フィルタ部オペレーション（FOP）から 1 件データを受け取ってはそのビュー部（V）が持つ処理を行う．ビュー部（V）は複数のフィルタ部オペレーション（FOP）からのデータを受け取ることが可能で，これにより複数のフィルタ部（F）に分

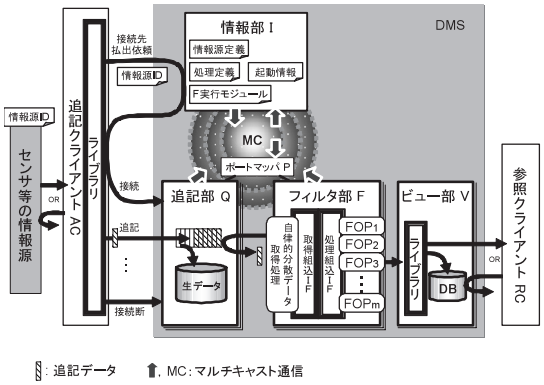


図 3 DMS 内でのデータの流れ
Fig. 3 Flow of data in the DMS.

散したデータをまとめなおすことが可能になる．そのためにデータ合流のためのバッファを持ち，バッファ内での合流・整列処理を行う．なお，ビュー部（V）は処理の C++ 組み込み I/F を持ち，ユーザ定義のビュー部（V）の組み込みが可能である．標準で組み込みのビュー部（V）としては PostgreSQL 接続ビュー部（V），および，一定量を越えたデータを高速に削除するためにリングバッファを用いたビュー部（V）がある．

以上のように，情報源で発生したストリームデータは追記クライアント（AC）→ 追記部（Q）→ フィルタ部（F）→ ビュー部（V）→ 参照クライアント（RC）と流れていく（**図 3**）．

3.4 情報部（I）

追記クライアント（AC）からの接続要求に応じて追記部（Q）の払い出しを行う．追記クライアント（AC）は払い出された追記部（Q）に接続を行う．これにより追記クライアント（AC）と分散する追記部（Q）との動的な対応付けを行う．情報部まで含めたデータの流れの処理を図 3 に示す．

情報部（I）は数十秒に 1 度の契機で系内のプロセス構成をマルチキャスト（MC）で共有し，新たに追加されたマシン上のポートマッパ（P）（3.5 節参照）に対して系への参加を促す．また http サーバを内包し，系内各マシンの起動情報や情報源情報，処理情報，スキーマ情報，最新の F 実行モジュールをファイルとして管理し，系内の各プロセスに http で公開する．なお，この情報部への参照負荷についてはロードバランサをフロントに設置することで対処する．情報部サービスが停止した場合でも，DMS はすでに接続済みのストリームデータへの処理は継続できるが，新しい追記部（Q）への接続払い出しと，マシン追加による新しいフィルタ部（F）プロセスの動的な系への参加についてはできなくなる．それを防ぐため情報部（I）は Active-Standby 構成により可用性を担保している．

3.5 ポートマッパ（P）

各マシンにはそのマシン内の追記部（Q）やフィルタ部（F）などの管理者としてポートマッパ（P）が 1 プロセス存在し，情報部（I），追記部（Q），フィルタ部（F）からときどき系内にマルチキャスト（MC）で流れる情報を受信し，系内に分散するプロセス構成の情報や追記部（Q）の輻輳状態を配下のプロセスに伝える．情報部（I）からの系内プロセス構成情報をポートマッパ（P）を経由して受信した追記部（Q）やフィルタ部（F）は，自らがプロセス構成情報に含まれていない場合に自らを含めるようにプロセス情報の修正依頼をマルチキャスト（MC）で系内の情報部（I）や各マシンのポートマッパ（P）に依頼することで系内情報の緩やかな最新化を図っている．

また，ポートマッパ（P）は情報部（I）の設定を参照して各マシン内のフィルタ部（F）や追記部（Q）の起動を実施する．よって，系に新しく参加するマシンは特に何も設定をせず，単にポートマッパ（P）のプロセスを起動するだけで系への参加ができる．情報部（I）の存在は系内に流れるマルチキャスト（MC）の情報から自動的に検出する．

フィルタ部（F）の動的機能入替えを管理するのもポートマッパ（P）である．フィルタ部（F）が追記部（Q）からデータを受け取る際に処理 ID の版番号が古いことを発見した場合には，マシンの代表であるポートマッパ（P）が F 実行モジュール入替えの処理を行う．すべてのフィルタ部（F）がいっせいに F 実行モジュール入替え作業に入ると系のサービスが停止してしまうため，DMS 管理者が指定した同時再起動数を超えないように再起動数の制御も行っている．この制御によって，1 つのマシンに多数のフィルタ部（F）が存在し，かつ，系が十分に大きく，多様な処理が混在する場合において，系の連続性を維持したまま機能の更新が徐々に行われていく．

4. 成長の具体的な実現

1 章で述べた，情報源，応用サービス，データへの処理内容，の各々の対象に対応して，多様化，追加，変更（含，削除）といった各成長を DMS がどのように支援し実現するかを説明する．

4.1 情報源の成長

4.1.1 情報源の多様化

多様な情報源の多様なデータ型への適応力を持って

いる．DBMS のレコードに相当するスキーマを持ち，データ型として int，long，double，char，varchar などの基本的な型および timestamp 型，blob 型，clob 型を持つ．clob 型を利用して 1 カラムからなるスキーマを使えば，実質的に XML への対応が可能である．

4.1.2 情報源の追加

3 章で述べたように，新たな情報源を追加する場合には，DMS 管理者が情報源情報を追加し，情報部（I）に配置し，対応する情報源 ID を追記クライアント（AC）側に情報源 ID を事前通知する．追記クライアント（AC）がその情報源 ID を使って DMS に接続すれば新たな情報源からの追記が可能になる．このとき，同時に走る他サービスを停止する必要がない．

4.1.3 情報源の変更・削除

前述の XML 対応にすれば XML タグの追加はサービス無停止で可能である．その場合，事前に新しいタグに対応する処理をフィルタ部（F）に追加しておく必要がある．ただ，追記データのスキーマへの動的なカラム追加はできず，その場合は，新たな情報源 ID に新しいスキーマを定義して追記クライアント（AC）からの再接続が必要になる．これは，追記クライアント（AC）側のデータ型変更を吸収するためには通常は追記クライアント（AC）の変更が必要であるとの考えからであり，強い制約にはならない．変更で特筆すべき点は，情報源の新データ型が旧データ型に変換可能であれば，フィルタ部（F）においてその変換を実施することで，情報源の変化をビュー部（V）に隠蔽することもできる能力を持つ点である．削除についてはデータが来ないことと等価であるため考慮は不要である．

4.2 応用サービスの成長

4.2.1 応用サービスの多様化

フィルタ部オペレーション（FOP）の 1 つである Emitter からデータを受け取ったイベントに応じて，そのデータに対して処理を行う callback 関数を定義するビュー部（V）組み込み I/F を提供している（3.3 節参照）．callback 関数として DBMS にデータを Insert するものを定義すればビュー部（V）として DBMS を動かすことができ，参照クライアント（RC）からの Select 要求に対応する Pull 型サービスが可能になる．また，callback 関数としてメールを送信するようにするなど Push サービスをビュー部（V）として利用することもできる．

4.2.2 応用サービスの追加

ビュー部（V）を新たに追加し，既存のストリームデータをそのビュー部（V）に流すためには，最初にビュー部（V）を起動し，そのビュー部（V）にデータを流す処理の処理 ID を追加し，既存の情報源 ID の設定にその処理 ID を追加すればよい．この際，既存の処理が停止することはない．追加されたビュー部（V）に情報を切り替えて配信するのではなく，複製して配信する場合にもフィルタ部オペレーション（FOP）の処理で対応が可能である．

4.2.3 応用サービスの変更

ストリームデータの送り先ビュー部（V）を変更する場合にはフィルタ部オペレーション（FOP）において配信先を切り替えることで対応可能である．

4.2.4 応用サービスの削除

配信先ビュー部（V）がなくなる場合でも，フィルタ部（F）で情報源に対して何もしないで廃棄するフィルタ部オペレーション（FOP）を使った処理 ID を指定し，追記部（Q）での生データ永続化フラグを ON にしておけば，生データは蓄積され続ける．さらに，後で追加されたビュー部（V）に対して情報源 ID と期間を指定して過去のデータを再度フィルタ部（F）を経由してビュー部（V）へ流し直すことも可能である．

4.3 処理内容の成長

4.3.1 処理内容の多様化

3 章で述べたようにフィルタ部オペレーション（FOP）を組み込む I/F を持つことで，SqlFilter や Emitter のほかにも，XML として流れてきたデータについて指定の条件を満たすか否かを判断したり，外部の DBMS プロセスや http サーバに問合せや参照を行ったり，文字列検出処理や映像処理を行ったりするなど多種多様な処理への適応力を持っている．フィルタ部オペレーション（FOP）の組み込みは追記部（Q）からデータを取得したイベントに応じて，そのデータに対して処理を行う callback 関数を定義することによって行う．処理情報でフィルタ部オペレーション（FOP）を指定する際に引数を持つことが可能であるため，その引数と，処理対象のデータを活用した処理を記述することができる．ストリームとして流れてくるデータに対して，その直前のデータとの差分を計算するなど過去のデータを利用して処理を行いたい場合には window 幅で指定された分の過去のデータを参照することもできる．

4.3.2 処理内容の追加・変更・削除

フィルタ部オペレーション（FOP）を新たに追加する場合には以下の手順をとる．

登録 step1 フィルタ部（F）のフィルタ部オペレーション（FOP）組み込み I/F に従って，新たなフィルタ部オペレーション（FOP）を組み込んだ

F 実行モジュールを作成する．

登録 step2 その F 実行モジュールを情報部（I）に登録（ファイルを置く）する．

登録 step3 新たに追加したフィルタ部オペレーション（FOP）の設定（名称と引数）を処理情報内に追加する．

登録 step4 処理 ID の版番号を上げる．

上記 step で系の一部に新しい機能の登録は行われたが，まだ系内に分散した各フィルタ部（F）の更新は行われない．そこで次にフィルタ部（F）が徐々に更新されていく処理の流れを示す．

入替え step1 フィルタ部（F）は自律的分散データ取得処理によって 1 つ追記部（Q）を決定し，その追記部（Q）上のキューからデータを情報源 ID とともに Pull する．フィルタ部（F）はその際，自らが持つフィルタ部（F）内の処理 ID の版番号を追記部（Q）に通知し，追記部（Q）は自分の持つデータに必要とされる処理 ID の版番号と比較し，版番号チェック結果も返却する．

入替え step2 フィルタ部（F）が追記部（Q）からデータを受け取る際に処理 ID の版番号が古いことを発見した場合には，そのデータの処理をあきらめて他のフィルタ部（F）に処理を譲り，マシンの代表であるポートマッパ（P）が F 実行モジュールを情報部（I）からダウンロードする．

入替え step3 ポートマッパ（P）はフィルタ部（F）プロセスの再起動による更新を行う．

入替え step4 フィルタ部（F）プロセス起動後，フィルタ部（F）が自律的に処理を開始する．

ここで追加・変更・削除の対象となる処理は情報源に対する処理，応用サービスのための処理のいずれであってもかまわない．さらに，処理情報内のフィルタ部オペレーション（FOP）系列の変更，標準提供されるフィルタ部オペレーション（FOP）を組み合わせ，新たな処理 ID を情報源情報内に追加することでの処理内容の変更も可能である．情報源情報に複数の処理 ID が定義できるため，特定の情報源の情報を複数の応用サービスで利用する場合に，それらの応用間の処理の追加や削除を処理 ID を単位として独立に行うことができる．

以上のように，DMS は，情報源，応用サービス，データへの処理内容，の各々の対象の各種成長に対応する．

5. 評　　価

3 章で述べたプロトタイプを使った評価について示す．

5.1 応用サービスへの適用

DMS のアーキテクチャが汎用的で幅広い応用に適用可能であることを実証するため 2 つの実験システムを構築した．

5.1.1 不正 SQL 検出システム

2006 年 6 月成立の日本版 SOX 法（金融商品取引法）によって産業界がデータベースへのアクセス監査の対応を急いでいる．一方，インターネット上のサービスに対して SQL-injection 攻撃（SQL 文字列の不正変更攻撃）などの新たな攻撃手法も広まりを見せている．それらに対応してアプリケーションサーバと DB サーバの間でやりとりされる SQL の情報の異常に対する即時検出処理が求められている．

DMS へのマッピング：

- 追記クライアント（AC）において，PostgreSQL のサーバに至るネットワークのパケットキャプチャを HUB のミラーポートを利用して行い，文字列（varchar 型）として DMS に投入する．
- 攻撃検出用のフィルタ部オペレーション（FOP）として grep に相当するものを用意し，ストリームとして流れてくる文字列に対してフィルタ部オペレーション（FOP）のパラメータで指定された SQL パターンを grep する．
- ビュー部（V）において端末に警報を表示する．

これらのマッピングに対応するプログラムを作成して動作を確認した．この構成によって，パケット内に流れる SQL-injection 攻撃の警報を出すことが可能になった．また，監視対象へのアクセスが頻繁であったり，不正検出処理の負荷が高かったりする場合であっても，DMS の特徴によって，マシン台数を増加させるだけで系の能力を高めることが可能である．

この不正パケット内文字列検出の構成は SQL 文に限らず，各種ビジネスシステム間の通信パケットやシステムログを追記部の入力として受け入れ，フィルタ部で不正利用パターンを検出し，ビュー部に警報を出力するものに汎用的に応用可能であり，監視について，監視対象（= 情報源）の各種成長，監視処理内容（= 検出ルール）の各種成長に適応可能なシステムになる．

5.1.2 映像監視システム

最近の凶悪犯罪の増加にともなって社会インフラとして監視カメラの整備が期待されている．また，インターネットに流れ込むセンサデータとして Web カメラは無視できない量の情報源になっている．

DMS へのマッピング：

- 追記クライアント（AC）として，Web カメラ

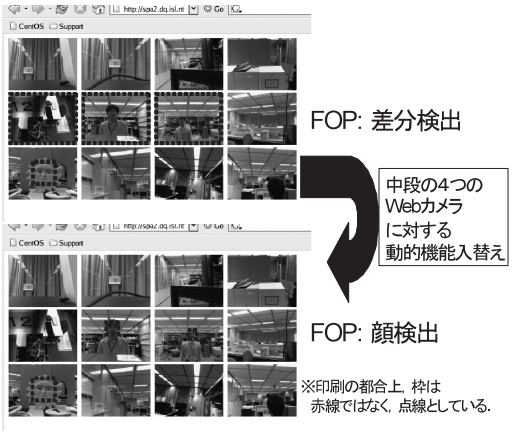


図 4 FOP の動的入替え例
Fig. 4 An example of FOP dynamic upgrade.

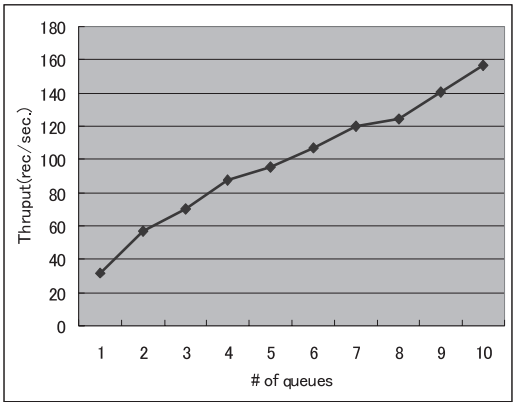


図 5 追記のスケールアウト
Fig. 5 Scale out of queue.

（Motion Jpeg）の各フレームを Jpeg 画像（blob 型）として DMS に投入する．

- 映像差分検出向けのフィルタ部オペレーション（FOP）群として，(1) 連続する 2 枚の Jpeg 画像の差分を比較し，閾値未満であれば，縮小画像を作成して画像一覧ビュー部（V）に転送するもの，(2) 閾値以上であれば縮小画像に赤枠を付与して画像一覧ビュー部（V）に転送するもの，(3) 閾値以上であればオリジナルサイズの画像を差分映像保存ビュー部（V）に転送するものを用意し，フィルタ部オペレーション（FOP）系列に (1)，(2)，(3) の順に定義し，実行する．
- 顔検出フィルタ部オペレーション（FOP）として (4) Jpeg 画像の中から画像解析[11] によって顔画像を検出し，検出した顔に赤枠を付与して画像一覧ビュー部（V）に転送するものも用意する．
- 画像一覧ビュー部（V）として Web ブラウザで多数の映像を一覧表示する．
- 差分映像保存ビュー部（V）として一定時間のオリジナル映像を保存し，検索により表示する．

これらのマッピングに対応するプログラムを作成して動作を確認した．この構成によって，Web カメラに対する差分検出の警報を出すことが可能になった．なお，映像監視のケースでは DSMS の特徴的機能である window を利用し，フィルタ部オペレーション（FOP）内で直前の Jpeg フレームを参照している．

図 4 は，映像監視システムの実行画面を上下に 2 画面分記載している．12 個の Web カメラ映像の 1 つ 1 つを情報源と考え，上段および中段の 8 情報源が映像差分検出（差分があった画像に枠をつける），下段の 4 情報源が顔検出（検出した顔に枠をつける）で動作している．さらに，その後の任意のタイミングで中段の 4 つの情報源に対する処理情報を映像差分検出から顔検出に無中断で変更した場合の画面を示している．このように，DMS は情報源に対する動的な処理の変更が可能になっている．

以上の 2 応用へのマッピングと動作確認によって，DMS のアーキテクチャおよび現プロトタイプが十分に汎用的で幅広い応用に適用可能な基礎的能力を持つことが確認できたと考える．

5.2 評価実験

DMS は 2 章で述べたように系無中断のスケールアウトと動的機能入替えの実現を狙っている．そこで映像監視試験モデルをベースに追記部（Q）とフィルタ部（F）のスケールアウト能力と情報源の追加や処理の変更にともなう系連続性の評価を行った．

【共通する測定環境】

- PC サーバ：CPU Pentium4 相当 3 GHz，RAM 2 GB
- OS：FedoraCore5
- 追記データ：PPM 画像（900 KB/枚）が 1 レコード
- スキーマ：blob のみ 1 カラム
- FOP：Magick++ を使って画像の差分を抽出
- ビュー部：なし（フィルタ部でデータ削除）

(1) 追記部（Q）の追加とスケールアウト

図 5 が追記部（Q）のプロセスおよびマシンを追加した場合の系のスループットを示している．

【測定環境】

- AC：2 台．プロセスが複数の場合はできるだけ均等に分配．各追記部（Q）に 200 レコードを投入．
- Q：1 プロセス/台で測定時に台数可変．データの永続化は ON，書き込み確認（sync）は ON．

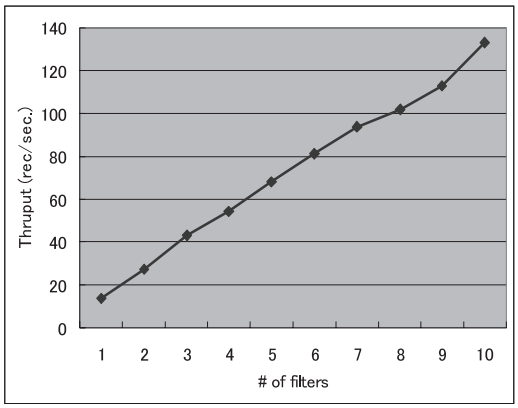


図 6 フィルタ処理のスケールアウト
Fig. 6 Scale out of filter.

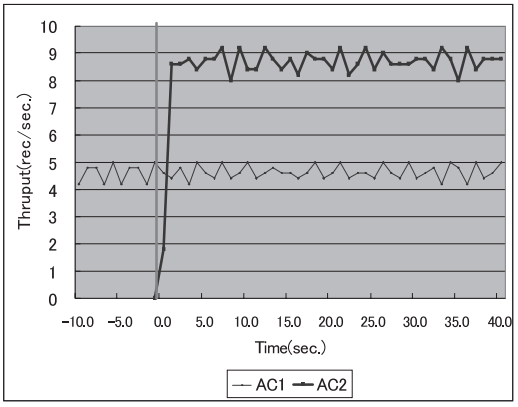


図 7 情報源の追加
Fig. 7 Source addition.

- F：フィルタ部（F）がボトルネックとならないために十分な台数（20 台．1 プロセス/台）とした．

【測定】

- Q：1 台〜10 台のそれぞれで測定．
- AC：追記部（Q）に対応する追記クライアント（AC）は追記部（Q）台数と同じプロセス数．
- 測定区間：追記クライアント（AC）で書き込みを始めてから終了するまでの時間を測定しスループットに変換．

追記部（Q）が系のボトルネックとなるのは，追記部（Q）のプロセス数が非常に多い場合，追記部（Q）の永続化負荷が非常に高い場合，追記部（Q）に対するフィルタ部（F）の Pull 要求が多すぎる場合などがある．今回は追記部（Q）の永続化負荷が高い場合を想定した性能測定を行っている．DISK 書き込み応答時間のバラツキはあるものの，グラフが示すとおり，追記部（Q）のマシン台数を増やすこと（同時に追記部（Q）プロセス数も増加）で，系のスループットが増加（スケールアウト）していることが分かる．

(2) フィルタ部（F）の追加とスケールアウト

図 6 がフィルタ部（F）のプロセスおよびマシンを追加した場合の系のスループットを示している．

【測定環境】

- AC：1 台．追記部（Q）に 1,000 レコードを投入．
- Q：追記部（Q）がボトルネックとならないために十分な台数（2 台．1 プロセス/台）とした．データの永続化 OFF．
- F：1 プロセス/台で測定時に台数可変．

【測定】

- F：1 台〜10 台のそれぞれで測定．
- 測定区間：追記部（Q）に最初のデータが到着してから，追記部（Q）上のデータがなくなるまでの時間を測定しスループットに変換．

グラフが示すように，フィルタ部（F）のマシン台数を増やすこと（同時にフィルタ部（F）プロセス数も増加）で，系のスループットが増加（スケールアウト）していることが分かる．なお，この図は横軸をフィルタ部（F）マシン台数にしているが，横軸を時間軸にして連続的なスケールアウトをさせることも可能であることを確認している．

(3) 情報源の追加と系連続性

図 7 は DMS の情報源と処理の追加にともなうスループットの変化を表している．

【測定環境】

- AC1：最大で 5 レコード/秒の追記（200 ms の wait）．
- AC2：最大で 10 レコード/秒の追記（100 ms の wait）．
- Q：各追記クライアント（AC）に各 1 台．1 プロセス/台．
- F：5 台．1 プロセス/台．

【測定】

- Step1：時間軸 −10.0 秒から情報源 1（AC1）の追記を開始．
- Step2：ある時点（時刻 0.0 秒）で情報源 2（AC2）の追記開始．
- 測定値：1 秒ごとにスループット値を測定．

グラフが示すように，新たな情報源の追加によって，追加以前の情報源に対する処理がほとんど影響を受けていないことが分かり，情報源の成長に対する系連続性が実現できているといえる．

(4) 処理内容の動的入替えと系連続性

図 8 は DMS の F 実行モジュールの動的な入替えにともなうスループットの変化を表している．

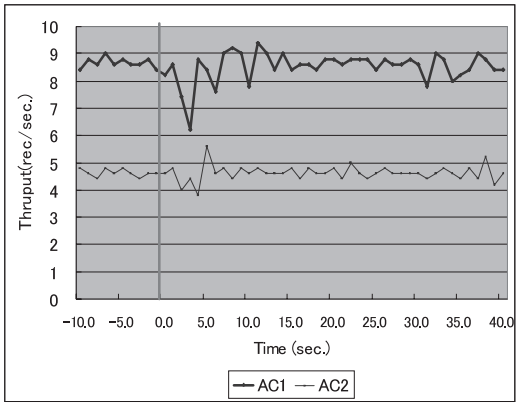


図 8 F 実行モジュールの入替え
Fig. 8 Exchange of filter module.

【測定環境】

- AC1：最大で 10 レコード/秒の追記.
- AC2：最大で 5 レコード/秒の追記.
- Q：各追記クライアント（AC）に 1 台．1 プロセス/台.
- F：2 台．5 プロセス/台.
- FOP1.0：PPM 画像の差分検出（赤枠版）
- FOP2.0：PPM 画像の差分検出（青枠版）．赤枠版と処理内容はほぼ同等である.
- F 実行モジュール 1.0：FOP1.0 を組み込んだもの.
- F 実行モジュール 2.0：FOP2.0 を組み込んだもの.
- P：フィルタ部（F）の同時再起動上限は 1 プロセス.

【測定】

- Step1：時間軸 −10.0 秒から AC1 および AC2 の追記を開始.
- Step2：ある時点で
  F 実行モジュール 2.0 を情報部に置き,
  処理情報内の FOP1 を FOP2 に変更し,
  追記クライアント（AC）に対する処理の版番号を 2.0 に変更し,
  管理コマンドで系内に通知を実施．これにより追記部（Q）の管理する版番号が追記部（Q）に通知される.
- Step3：フィルタ部（F）は自律的なデータ取得時点で版番号の差異を発見し，ポートマッパ（P）を経由して F 実行モジュールの入替えを実施する．その入替え後の F 実行モジュールの実行が開始されたタイミングを時刻 0.0 秒としている.
- 測定値：1 秒ごとにスループット値を測定.

F 実行モジュールの動的な入替えの際には，フィルタ部（F）プロセスの部分的な再起動が発生しているが，この AC1 のグラフが示すように，若干のスループットの変動は見られるものの，大きな変動は見られない．また，同時に動作している AC2 からの処理に関してもスループットへの影響がほとんど見られない．これらから，処理の成長に対する系連続性が実現できているといえる.

6. 関連研究と今後の課題

6.1 関 連 研 究

すでに 2 章で DMS の機能成長や規模成長などの面について類似技術との比較を行ったが，そのほかの面での関連研究について述べる.

ハードウェアリソースに着目し，処理負荷への追従，あるいは，未知なものを含めた障害すら乗り越えて生き延びさせる性質を持つシステムは，小磯らにより研究されており，自律した複数の構成要素が連合することでシステムを構成し，互いの役割を補完しあうことによって不測の障害を乗り越えていく自律連合型システムが提案されている[12]．小磯らは可用性向上に重点をおいた仮想マシン上でのリソース管理を実現しているが，本論文ではシステムの成長に向けた，機能の追加・世代交代におけるシステム負荷変動の吸収手段が必要であり，このメカニズムに重点をおいた提案を行った.

DMS は Google File System[13] にいくつかの点で影響を受けている．それは，格安の PC マシンを多数並べて全体をスケールアウト構成にする点，多少のマシンの停止では系が停止しない可用性を実現する点，マシンの追加など運用コスト削減を狙う点，である．また，データへの処理環境という点から見ると Google File System 上で動作する MapReduce[14] とも近い．しかし DMS はセンサ・ビジネスの多様性を考えた各種連携の仕組みを備え，ストリームデータへの即時処理が行え，DSMS として機能（Continuous Query）を持つ点が本質的に異なる.

適合型ネットワーキングプラットフォーム Ja-Net[15] は，ユニバーサルネットワーク上で利用可能なコンテンツ・ソフトウェア・デバイスを組み合わせてユーザの嗜好や行動パターンに応じて動的にサービスを生成する適応型ワットワークアーキテクチャを提案している．しかし，我々の考えるサービス成長にともなうスケーラビリティへの対応や，データストリーム管理システムとしての機能整備は行われていない.

広域で処理を分散し可用性も高める技術として P2P がある．たとえば PIAX[16] は，ユーザの位置や情報間の関係に基づくオブジェクトの発見と連携をスケー

ラブルに実現する PF であり，狙いとするユビキタスサービスのイメージは DMS のサービス適用イメージとも近い．現在の DMS はセンタ内に閉じている構成だが，今後は P2P 技術などとの連携した広域対応のあり方も考えていきたい．

6.2 今後の課題

1 章で述べたとおり，我々の目指す情報統合管理サービス PF において，効率的な半永久保存性に関しては，今後の研究の中で解決を図って行きたい．

また 2 章でもふれたが，システム全体を継続したまま，ソフトウェアの世代交代を行うためには，あらかじめシミュレーションなどによって機能追加や削除の影響を把握する手段が必要である．すでに工業プラントの例[17] では，シミュレーション技術を駆使して事前に整合性を確認した後に組み込みを行う技術が確立しているが，汎用的なソフトウェアである DMS 上での技術確立も，今後の重要な課題の 1 つである．

そのほか，フィルタ部（F）の完全性実現にともない，フィルタ部（F）内のフィルタ部オペレーション（FOP）数が大規模化する問題に対応するためのフィルタ部（F）のグループ化，不正なフィルタ部オペレーション（FOP）の動作に対応する技術，NW 帯域の利用のさらなる効率化，ビュー部（V）の成長の実現，異種ストリームの合流なども現在の設計では十分なレベルに達していないため今後の課題として残る．また，より効率的な自律分散データ取得アルゴリズムの開発も継続して行っていく．

7. ま と め

時々刻々と生成される多種多様なセンサからの追記型のデータに対して，適切に蓄積・管理するとともに，利用者の目的に合った形で情報を統合し，検索や分析といった参照を可能にするプラットフォームの構築に向けて，ストリームデータを対象とし，データの流れ方向と処理量の増加方向に対する 2 つの分散軸を持ち，系としての連続性を維持したまま機能的成長と規模的成長を実現する追記・参照型 DMS アーキテクチャを提案した．そのアーキテクチャに基づいたプロトタイプを構成し，情報源，応用サービス，データへの処理への各種成長の実現方式を述べ，複数の実験システムを構築することで多様な応用への可能性を示し，機能成長と規模成長に必要となる系無中断のスケールアウトの能力と，連続データに対する系無中断の動的機能入替えについて評価により確認した．

謝辞 「映像からの動き・顔検出コアプログラム」の提供および組み込みにご協力いただいた NTT サイバースペース研究所新井啓之氏，小島明氏に謹んで感謝します．

参 考 文 献

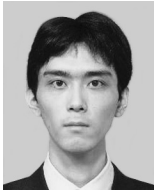
1) 喜連川優ほか：研究領域「情報爆発時代に向けた新しい IT 基盤技術の研究」. http://itkaken.ex.nii.ac.jp/i-explosion/
2) 赤間浩樹，内山寛之，三浦史光，西岡秀一，内藤一兵衛，谷口展郎，山室雅司，櫻井紀彦：追記・参照型データ管理プラットフォームアーキテクチャの提案，情報処理学会 DPSWS 2006, pp.199–204 (2006).
3) 内山寛之, 赤間浩樹, 西岡秀一, 内藤一兵衛, 谷口展郎，長谷川知洋，三浦史光，山室雅司，櫻井紀彦：分散データストリーム処理アークテクチャの提案，情報処理学会研究報告 2007-DBS-143, DBWS 2007, pp.327–332 (2007).
4) Bourke, T.: サーバ負荷分散技術，O'REILLY (2005).
5) The STREAM Group: STREAM: The Stanford Stream Data Manager, *IEEE Data Engineering Bulletin*, 26-1 (2003).
6) Abadi, D., Carney, D., Cetintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N. and Zdonik, S.: Aurora: A New Model and Architecture for Data Stream Management, *VLDB Journal*, Vol.12, No.2, pp.120–139 (2003).
7) Johnson, T., Muthukrishnan, S., Spatscheck, O. and Srivastava, D.: Streams, Security and Scalability, *Keynote in Proc. 19th Annual IFIP Conf. on Data and Applications Security*, Lecture Notes in Computer Science 3654, pp.1–15, Springer-Verlag (2005).
8) Cherniack, M., Balakrishnan, H., Balazinska, M., Carney, D., Cetintemel, U., Xing, Y. and Zdonik, S.: Scalable Distributed Stream Processing, CIDR (2003). http://www.cs.brown.edu/research/aurora/cidr03.pdf
9) 渡辺陽介，北川博之：仮想マシン技術を用いた持続型ストリーム処理環境の評価，情報処理学会研究報告 2007-DBS-143, DBWS2007, pp.339–344 (2007).
10) 山田真一，渡辺陽介，北川博之：ストリーム処理とデータベースを統合した実世界情報管理基盤，電子情報通信学会 DEWS2006 5C-i7 (2006).
11) 新井啓之，磯　和之，小島　明，仲澤　斉，小池秀樹：インテリジェントな映像モニタリングを目指して，NTT 技術ジャーナル 2007.8, pp.9–12 (2007). http://www.ntt.co.jp/RD/OFIS/active/2006pdf/hot/ap/04.html
12) 小磯知之，安部洋丈，池嶋　俊，石川宗寿，リチャード・ポッター，加藤和彦：サステーナブ

ルサービスのための基盤ツールキットの設計，情報処理学会論文誌：コンピューティングシステム，Vol.48, No.SIG3(ACS17), pp.13–26 (2007).
13) Ghemawat, S., Gobioff, H. and Leung, S.: The Google File System, *ACM SOSP2003*, pp.29–43 (2003).
14) Dean, J. and Ghemawat, S.: MapReduce: Simplied Data Processing on Large Clusters, *OSDI'04*, pp.137–150 (2004).
15) 川西 直，板生知子，森川博之，青山友紀：適応型ネットワークアプリケーションのための類似性に基づいた自律分散コンポーネントの発見方法，情報処理学会 DICOMO2002, pp.205–208 (2002).
16) 吉田 幹，寺西裕一，春本 要，下條真司：マルチオーバレイと分散エージェントの機構を統合化した P2P プラットフォーム PIAX，情報処理学会研究報告 DPS2006-DPS-128, pp.43–48 (2006).
17) 三浦真太郎，横山克己：OmegaLand 開発コンセプトと機能概要，横河技報，Vol.45, No.1, pp.63–66 (2001).
18) 川島英之，今井倫太，遠山元道，安西裕一郎：センサデータベースシステム KRAFT の設計と実装，情報処理学会論文誌：データベース，Vol.45, No.SIG14(TOD24), pp.39–53 (2004).
19) 渡辺陽介：データストリーム処理システムに関する研究動向，電子情報通信学会 DEWS2004 ミニサーベイ (2004).
20) 白石 陽：センサネットワークのためのデータベース技術，情報処理，Vol.47, No.4 (2006).

(平成 19 年 5 月 17 日受付)
(平成 19 年 11 月 6 日採録)

赤間 浩樹（正会員）

1990 年東海大学大学院理学研究科数学専攻修士課程修了．同年日本電信電話株式会社入社．以来，ネットワーク向け DBMS，ニュース・オン・デマンド，マルチメディア情報検索の研究開発，FLET'S 系 PF の実用化等を経て，現在，情報統合管理技術の研究に従事．日本データベース学会，人工知能学会，ACM 各会員．



内山 寛之

2002 年大阪大学大学院基礎工学研究科修士課程修了．同年日本電信電話株式会社入社．現在，分散ストリーム処理システムの研究開発に従事．日本データベース学会会員．

西岡 秀一（正会員）

1995 年横浜国立大学工学部電子情報工学科卒業．同年日本電信電話株式会社入社．以来，データベース管理システム，著作権管理システム，XML 処理システムの研究開発に従事．博士（工学）．日本データベース学会会員．

内藤一兵衛

2006 年早稲田大学大学院理工学研究科情報・ネットワーク専攻修了．同年日本電信電話株式会社入社．データベース，情報検索等に興味を持つ．日本データベース学会会員．

谷口 展郎（正会員）

1994 年東京大学大学院工学系研究科機械工学専攻修士課程修了．同年日本電信電話株式会社入社．以来，著作権保護，画像検索，プライバシ保護等の研究開発に携わる．現在，同社サイバースペース研究所において分散データ処理に関する研究開発に従事．

長谷川知洋

1997 年筑波大学大学院理工学研究科修了．同年日本電信電話株式会社入社．XML 索引技術，高可用化データベース，分散環境における半永久持続システム等の研究開発に従事．

兵藤 正樹（正会員）

2004 年北陸先端科学技術大学院大学知識科学研究科修士課程修了．同年日本電信電話株式会社入社．XMLDB における検索技術の研究に従事．

三浦 史光（正会員）

1990 年京都大学大学院工学研究科修士課程修了．同年日本電信電話株式会社入社．セキュリティプロトコルおよび並列処理に興味を持つ．

山室　雅司（正会員）

1987 年早稲田大学大学院理工学研究科数学専攻修士課程修了．同年日本電信電話株式会社入社．1990 年コロンビア大学大学院電気工学研究専攻修士課程修了．以来ネットワーク設計法，データベース設計・可視化，マルチメディア情報検索，デジタル情報流通基盤の研究開発に従事．博士（工学）．1994 年電子情報通信学会学術奨励賞．電子情報通信学会，日本ソフトウェア科学会，日本データベース学会，IEEE-CS 各会員．情報処理学会情報規格調査会理事．

櫻井　紀彦（正会員）

1979 年早稲田大学理工学部電気工学科卒業．同年日本電信電話公社（現 NTT）入社．ファイル記憶階層アーキテクチャ，データベース応用技術，コンテンツ流通管理技術，セキュリティ技術の研究開発に従事．電子情報通信学会会員．