4N-1

エージェント技術を適用したリアルタイムデータ集計処理の設計

村田　悠也†　　山田　隆志†　　山本　学†　　吉川　厚†　　寺野　隆雄†
†東京工業大学

1　はじめに

ブログ，SNS 等の Web サービスやスマートフォンの普及に伴い，処理するデータは年々増加し，海外では秒間数十万〜百万件に達するシステムが確認されている．このデータ発生件数は，業務システムで利用されてきたデータ蓄積型のバッチ処理では，処理が困難でありスケールアップやスケールアウトを必要とする．スケールアウトによるバッチ処理では，Hadoop[1]，グリッドバッチといった分散バッチ処理技術が存在する．しかし，分散バッチ処理で行われる集約処理といった高速化に関わる設計は難しく，台数の増加に伴い設計の難度が上がる．今後もデータの増加が予想され，システムの高速化と高速化設計の簡易な設計手法が望まれる．

一方，リアルタイムデータ集計処理 (Realtime Data Aggregation, RDA) という手法がある [4]．これは，ストリーム処理技術とデータストア技術を組み合わせた処理手法で，高速・高スケーラビリティという性質を持つ．すでに，高スケーラブルかつ高速なストリーム処理技術が提案され，実現されている [2]．しかし，ストリーム処理技術には，データストアの機能を持たないか，直近のデータを保持する機能しかない場合がほとんどである．一方，従来のインメモリ技術よりも数倍から数十倍高速なエージェントプログラミングモデルに基づくインメモリデータストアが提案されている [3]．このデータストアは，データをインメモリで保持する機能に加え，エージェント間のメッセージング機能やパイプライン・並列処理を実現する機能を持ち，すでに高速な RDA の実現可能性が報告されている [5]．エージェント技術を適用した RDA では，エージェント設計がシステム設計に影響し，マルチエージェントの構成がシステムの高速化に影響する．[5] で提案されたシステムでは，エージェント設計，分散配置等の設計方法論は存在せず，技術者に依存する設計となっている．

本稿では，エージェント技術を適用した RDA システムの設計手法を提案し，その手法によってマルチエージェントシステムによる RDA が設計できるか検証する．提案手法では，要件からエージェントを導き出すエージェント設計と設計された多数のエージェント群を複数の計算機群に適切に配置するエージェント分散配置の 2 つの工程から成るが，本稿では，エージェント設計に関して述べる．エージェント分散配置に関しては，まだ未解決な点が多く，今後の課題とする．

Design of Agent-based Realtime Data Aggregation
†Yuya Murata　†Takashi YAMADA　†Gaku YAMAMOTO
†Atushi YOSHIKAWA　†Takao TERANO
†Tokyo Institute of Technology

2　マルチエージェントシステム

本稿で述べるエージェントは，自分だけがアクセス可能なデータを保持し，メッセージを受けて，そのデータにアクセスしながら処理を遂行するソフトウェアエンティティである [5].

各エージェントは，識別子 (キー)，メッセージハンドラ，処理ロジック，データ (レコード) を持つ．アプリケーションは，エージェントの生成・削除，メッセージ送信ができる．メッセージを受信したエージェントは，メッセージ処理により管理するデータの更新・参照を行う．エージェントはデータをレコード型として持ち，少なくとも 1 つのレコード (マスタレコード) を持つ．また，マスタレコードをルートとするツリー状のレコードセット (複数のレコードを管理するオブジェクト) を設定することも出来る．ひとつのエージェントは，同時並行で複数のメッセージを処理することはなく，エージェントのメッセージ処理は，ひとつのトランザクションとして処理される．

3　RDA システムの設計

RDA システムのエージェント設計では，1) システム要件を定義する要件設計，2) システムを設計するデータレコードと処理の関連付け設計，3) システム中のエージェントを設計するエージェントタイプの設計，4) エージェントタイプのインスタンス化の 4 ステップにより設計する．

本稿で例題として示す RDA システムは，ユーザーの位置情報 (GPS データ) からジョギング距離を算出し，リアルタイムにランキングするジョギング・ランキングシステムである．ランキングは，予め年齢や性別等のランキング表が用意されており，ユーザーはそれぞれのユーザー属性から参照するランキング表を決定する．また，ランキング表は，常に更新されユーザーはいつでも最新のランキングを参照できる．

3.1 要件設計

ジョギング・ランキングシステムの要件設計を行う．要件設計では，要求をだすユーザーと要求を処理するシステムの関係から，その動作を簡単なシナリオとして記述する．ジョギング・ランキングシステムでは，以下のようなシナリオを記述した．

1. ユーザーは，システムにユーザー情報を登録する．
2. ユーザーは，位置データをシステムに送る．
3. システムは，ユーザーごとに，位置データから距離を計算する．
4. システムは，ランキング種別ごとに，ユーザーの走った距離をランキングする．

記述したシナリオは，情報のフローとシステム構成を表現した概念設計となっている．

3.2 データレコードと処理の関連付け設計

要件設計を用いてシステム設計を行う．設計手法は，エージェントを導出しやすくするために「データレコードと処理の関連付け設計」を行う．要件から入力データ，出力データを取得し，入力と出力を処理によって繋ぐ．その際，処理中で使われるデータはレコード型とし，レコード列が変化する処理を記述する．レコード列が変化しない処理は，次の処理にまとめる．レコードの設計では，主体となる識別子 (主キー) をシナリオから読み取り，そのキーから参照されるデータのうち処理で使うものをレコードとする．同じキーならば前のレコードと結合し新たなレコードとする (図 1).

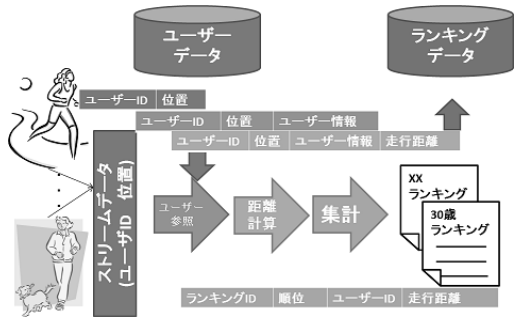

図 1: ジョギングランキングシステム

3.3 エージェントタイプ設計

図 1 を用いて，エージェントタイプ設計を行う．エージェントタイプは，同じ主キーのレコードと関連付けられた処理を持ち，異なる主キーのエージェントタイプと通信する．図 1 では，ユーザー ID を主キーとするレコードとランキング ID を主キーとするレコードがあるため，図 2 のようにエージェントタイプは 2 種類となる．

3.4 エージェントタイプのインスタンス化

エージェントタイプのインスタンス化は，各エージェントタイプが持つ主キーによってインスタンス化される．図 2 のエージェントタイプの主キーを用いて，ユーザーエージェントタイプはユーザー ID，ランキングエージェントタイプはランキング ID によって図 3 のようにインスタンス化される．

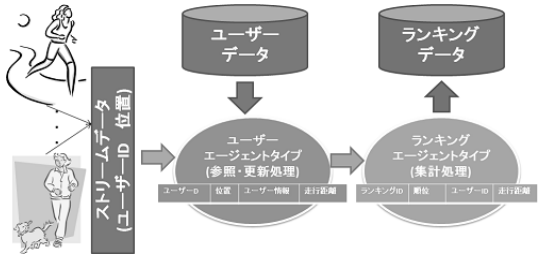

図 2: エージェントタイプ設計

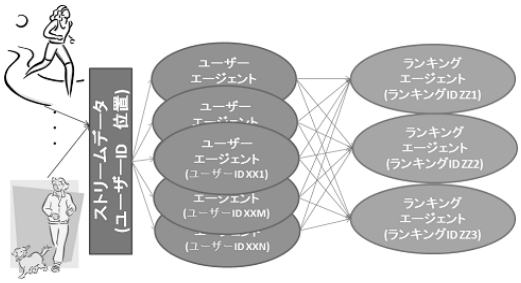

図 3: エージェントタイプのインスタンス化

4 まとめ

本稿では，RDA システムのエージェント設計を 4 ステップにより設計する方法を提案し，ジョギング・ランキングシステムを例題として提案手法による設計を行った．エージェントタイプが 2 種類，処理の分岐なしと単純な例であるが，エージェントプログラミングモデルを用いたシステム設計で，エージェントプログラミングモデルに精通した設計者が経験的に似たような方法で設計していることがわかっている．本稿で示した手法は，まだ不十分な部分があるが，この手法の確立により高速処理が必要な RDA 設計の技術レベルをを低くすることが可能となる．また，RDA 以外のアプリケーションの適用も考えられ，設計が非常に難しい高速トランザクション処理システムの設計を容易にすることも期待できる．残されている技術課題として，エージェントを複数の計算機にどのように分散配置するかという分散配置設計の問題が残されている．

参考文献

[1] http://hadoop.apache.org/ , 2011
[2] http://www-01.ibm.com/software/data/infosphere/streams/ , 2011
[3] 山本学:エージェントプログラミングモデルの高速トランザクション処理システムに対する効果の考察, in Joint Agent Workshops and Symposium(JAWS) (2008)
[4] 山本学： エージェントプログラミングモデルに基づくデータキャッシュの性能評価, 合同エージェントワークショップ＆シンポジウム 2010 (2010)
[5] 山本学：高速マルチエージェントシステムによるリアルタイムデータ集計処理の考察, in Joint Agent Workshops and Symposium(JAWS) (2011)