

CAD 分散環境のためのワークフローアプローチ

中村美枝† 中島健造‡ 小西 修‡

1 企業におけるワークフローとしてではなく、多企業間での情報の共有を図り、分散 CAD データベースをバックエンドとしたプロジェクト管理まで行なうワークフローの有り方を提案する。

各プロセスの制御は、DB に持たせた制御データと KQML によるエージェント間通信を用い、CORBA による分散オブジェクト間の企業間非同期通信により実現する。XML と Java をユーザインタフェースとすることで、柔軟性が向上する反面 XML パーサによる処理コストも増大する傾向にある。トランザクションの発生が、数時間単位であるようなシステムを例に挙げ、ビジネス・ロジックの実装イメージを展開する。

The Workflow Approach for Distributed CAD Environment

Mie Nakamura†, Kenzou Nakajima‡, and Osamu Konishi‡

In this paper, we describe an agent based workflow management system where the workflow engine is implemented in an agent system with ECA rules. The focus here is on the architectural approach and on the prototype system of a distributed workflow system including the CAD data. We define an agent-based framework for accessing heterogeneous resources at Web sites. KQML-CORBA together with the Web is used as the distribution infrastructure. The CORBA methods on the server are invoked through XML. We realize the communication of agents in the system through a KQML implementation based on CORBA. Agent-based computing advances database transaction and control flow management concepts and remote programming techniques.

1. はじめに

製造業における中小企業は、大企業の東南アジアや中国への工場設立を受けて利益の縮小や納期の短期化など厳しい状況におかれている。従来の受注を受けて納品する請負型では、現状からの逸脱は困難である。しかし、その一方で世界に通用する製品を作り出している中小企業もある。個々の企業のポテンシャルを高め、活性化することが起死回生の切り札である。

各社が得意分野を活かし、分散化された環境

で製造を行い、それらを融合することで新しい製品の開発・製造・販売を実現することが可能となる。しかし、そのためには、企業間連携を密にし、リアルタイムに作業進捗を把握する必要がある。

今回のシステムは、CAD 分散環境システムにおいて、情報の共有化を図り、生産効率を向上させるためにワークフローの利用を提案する。従来の企業内や部門内でのワークフローとは異なり、インターネットを介して XML による各社間のアプリケーション連携 (A to A) と Web サーバと CAD データベースのバックエンドシステムによる協調をミドルウェア基盤として採用する。これは、KQML を使った ECA エージェント間のプロセス制御と

† 岡山大学大学院自然科学研究科
Graduate School of natural science and
technology Doctor's Course, Okayama
University

‡ 高知大学理学部数理情報科学科
Department of Information science, Kochi
University

CORBA を用いた分散オブジェクト間の非同期通信により実現する。また、CORBA のポータビリティを活かし、システムを構築するために、ネットワーク接続は、インターネット、VPN や無線 LAN 等いずれにも対応可能でなければならない。IDL によるインタフェースに加え、リモート環境にあるエンドユーザとは、XML で作成された Web ページを介してコミュニケーションする。CAD データなどの設計の段階で作成された情報から製造に必要な情報を認識・抽出し、その情報に基づいて製造された部品は、ワークフローの記述に基づき、次の過程へと遷移する。

ビジネス・ロジックの展開例として、4 社からなる企業連合体が、ワークフローを利用した営業・製造・販売までの一貫した製造工程管理と CAD 分散データベースを挙げる。

2. 基本システム設計

ワークフローモデルにおける協調作業を実現するには、(1) 複数の知的作業員 (エージェント) が会話を交すための機構、(2) 知的作業員 (エージェント) の間で共有、変換される情報の蓄積、管理機構が必要とされる。

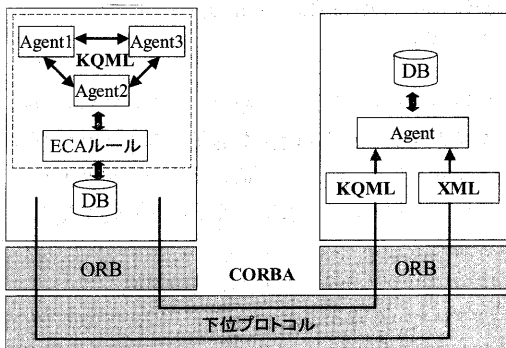


図 2.1 エージェントベースアーキテクチャ
Fig.2.1 Agent based architecture.

我々は、この二つの機構を、エージェントベースとデータベースの融合によるワークフローモデルを提案する。特に、各タスクを実行するエージェント間の協調促進を図るアクティ

ブ・ワークフロー・エンジン (以降、AWFE と記す) を有したアクティブ・ワークフロー・システムとして設計している。(図 2.1)

2.1 分散環境におけるワークフローモデル

このワークフローシステムの仕事は、ユーザの要求を受けて、いくつかのデータベース (情報源) へアクセスし、最適な結果をユーザに返すというプロセスとみなせる。このプロセスを成し遂げるために、組織化されたタスクの集まりがあり、各タスクを実行する作業主体をエージェントと見ることができる。我々のシステムでは、個々のタスクを複数のエージェントのグループが担当する。各エージェントは、タスクの特定部分を担当し、その部分に精通する。タスクは、その仕事を成し遂げるために、タスク全体の知識を必要とし、全エージェントが全体の知識を共有しながら実行される。このような環境では、エージェント間の行動に関して、(1) データ一貫性に影響を与えるような事態が生じた場合にエージェントに通知する告知機構や、(2) 密接に情報交換を行なう必要があるエージェントグループを扱うグループ機構が必要である。我々は、この視点から、タスクとエージェントの状態 (ステータス) に注目して、エージェント協調を ECA ルールで記述する方法を提案している。

2.2 エージェント間の協調

エージェントベースは、プロセス集約的であり、一方、データベースは、データ集約的であるといえる。この各々のアーキテクチャの性格を考慮すると、エージェントシステムには、一貫性制約や障害回復のためにトランザクションを取り入れる必要があり、また、アクティブデータベースシステムには、長時間プロセスとそれに関連した維持機能が必要である。この二つのパラダイムを統合するモデルとして、次のような表現が導かれる。

agent:
attributes: a set of local data
methods: methods
(local/access/composite)

lifecycle: structural sequence of methods
rules: structural dependency of methods
transaction: methods (commit/abort/
pre-commit/submit)

ここに、ポテンシャルな振る舞いとしてエージェントが実行できるアクション (メソッド) がある。このメソッドは、エージェント自身のローカルデータを操作するローカルメソッド、他のエージェントと通信するアクセスメソッド、そして、これらを組み合わせたメソッドからなる。ライフサイクルは、そのアクションが可能な実行シーケンスであり、ルールでは、エージェントのメソッド制御可能な状態間の構造的な従属関係を記述する。ここで、状態はメソッドの制御可能な状態を示し、トランザクションにおける、commit / abort / pre-commit / submit のどれかをとる。

エージェントの仕様あるいは実行時に制御されるエージェントのライフタイムの細粒度は、完了 (コミットかアボートの状態)、また、メソッドのサブミット (開始状態) である。アクセスメソッドの実際の実行時は、対応するデータベースあるいは協調するエージェントの制御の下にあるということである。さらに、いくつかのエージェントは、アクセスメソッドが完了しており、その結果は、パーマnentになる予定であるということを示すところの **prepare-to-commit** を提供する。ローカルメソッドに関しては、エージェントはまた、これらの実行時間上の制御を持つ。ここで、commit は、実行結果が、意味的に失敗か、意味的に成功かという二つの型を有する。

アクティブ・ワークフロー・システムを構成する各エージェントは、ユーザからの質問に答えるために、互いに協調しながら作業を進める。このとき、各エージェントの調停を行い、協調を促進させる役割を担う AWFE は、各エージェントの状態を管理しなければならない。そのため、各エージェントは 処理依頼を受けたり、処理を実行・終了したりと、何らかの状態変化

が起こる度に、AWFE に自分の状態を告知する。これにより、各エージェントは AWFE を介して、協調相手の状態を知ることが可能となり、より豊かなエージェント間の協調が行なえる。

また、このシステムにおけるのエージェント群の振る舞いは、次の2点で制約される。

- (1) エージェントは、自己の他へのアクションについては、必ず AWFE に問い合わせねばならない。AWFE は、それに必ず答えなければならない。また、AWFE は問い合わせという事実がないのに、応答することは禁止される。
- (2) エージェントは、AWFE から紹介された他のエージェントへのみ、要求をすることが許可される。

具体的にエージェント群の振る舞いの流れを説明する。

- (1) 情報源エージェントは、ドメイン制約も含めたドメイン情報を AWFE に通知する。
- (2) ユーザエージェントからの質問が、アクセスエージェントに送られる。
- (3) アクセスエージェントは、その質問を AWFE に送る。
- (4) AWFE は、各ドメイン情報と質問のマッチングをとり、最適の情報源エージェントをアクセスエージェントに知らせる。
- (5) アクセスエージェントは、その情報源エージェントに質問を發し、結果を受け取る。

ここに、AWFE は、各エージェントの行動 (生成、起動、休止、消滅など) やエージェント間のメッセージ転送を把握して、それらの活動の促進を図る。

2.3 エージェントベーストランザクションモデルと構造

分散環境におけるエージェントベースモデルにトランザクションの概念を導入する。

ここでのエージェントベースモデルは、情報エージェントと他のサイトにある情報源エージェントの間の相互作用（協調）という環境を想定している。

この情報エージェントのゴールは、質問を発して、情報源エージェントにソースデータベース（例えば、CAD データベース）の検索を依頼し、質問に対する結果を得るというものであり、そのタスクは、依存関係を持ったサブタスク群からなると考える。

これは、入れ子型トランザクションとして表現されるが、相手へのトランザクションの委譲 (delegation) やトランザクション間の構造的な依存関係などを考慮する必要がある。

そのため、拡張トランザクションの各ステップの実行を breakpoint や delegation そして dependency rules などの概念を導入することによって、トランザクションの実行を制御する方法を考える。

トランザクションの構造：

```

Transaction  get_results
  subtransaction  query
  subtransaction  search
  subtransaction  get_results
  integrity-constraints
end-transaction

task query_execute
  subtask      query
  subtask      search
  subtask      get_results
  constraints:
    query BEFORE search
    search BEFORE get_results
  Goal { results }
  Exit { cancel_get_results }
end_task

```

拡張トランザクションモデルでは、サブトランザクション間の相互作用についての構造を定義する。サブトランザクションは、それらの状態推移間の依存関係によって表現される。例えば、入れ子型トランザクションでは、もし、

親のトランザクションがアボートすれば、子のトランザクションもまた、アボートされなければならない。そのような従属性は、メソッド間の構造的な依存関係として、次のようなアクティブな依存ルールで記述できる。

```

Trigger : On Method i, Method j;
          If Method i enters State i;
          Do Method j must enter State j;
Order : On method i, Method j;
        If Method j has entered State j;
        Do Method i can(cannot) enter
          State i;

Real-Time : On Method i;
            Do submission Method i(); or
            completion Method i();

```

ルールは、指定された状況におけるエージェントの実際のアクションであり、アクティブに振る舞うルールである。

2.4 アーキテクチャと ECA ルール

各エージェントの協調を促進する AWFE は、各エージェントの状態をアクティブデータベースに更新することによって管理する。この時、アクティブ・データベースには、エージェントの状態を記述する以下に示すような二つのテーブルが必要である。

Agent [name,state]

もう一つのテーブルはエージェントの動向を逐一記述するための、いわゆるログであり次のようなテーブルである。

Agent_log

[name,operation,from,to,ope_state]

このテーブルはエージェント name において、処理 operation をエージェント from から依頼された、またはエージェント to へ依頼して、その処理の状態が ope_state であることを表現している。

このエージェントの状態を管理する二つのテーブルと ECA ルール機構を用いることにより、一貫性制約が表現出来る。

例えば、ユーザから処理依頼メッセージをユ

ユーザ・インタフェース・エージェントが受け取ると、アクセスエージェントにその処理を依頼し、最終的にはその結果をアクセスエージェントから受け取ってユーザに結果を提示する。この一連の流れの中には当然 AWFE が各エージェント間の調停を行い、一貫性が保たれないような状況の時はその原因であるエージェントを停止させたりする。ここでユーザ・インタフェース・エージェントのメソッド間の構造的制約は次のようなアクティブ・ルールで記述できる。

```

getQuery(query)-->sendQuery(search)
-->getResults(get_results)
Lifecycle: getQuery,askURL,getURL

Dependency rules:
trigger:
  On getQuery,sendQuery
  if getQuery enters state(commit)
  Do sendQuery must enter state(submit);
order:
  On getQuery,sendQuery
  if getQuery enters state(abort)
  Do getQuery cannot enter
                                state(submit);

real-time:
  On getResults
  if sendQuery weak-Begin-on-condition
  -dependency results=true
  Do submission getResults;

```

さらに実際に使用しているルールを用いた場合のエージェント状態管理テーブルのログを以下に示す。

表 2.1. 各エージェントのログ

Table 2.1 Log of agents.

name	operation	from	to	ope_state
UIAgent	&getQuery	&user	&UIAgent	&commit
UIAgent	&sendQuery	&UIgent	&AAgent	&commit
AAgent	&getQuery	&UIAgent	&AAgent	&commit
AAgent	&askUrl	&AAgent	&MAgent	&submit

2.5 KQML-CORBA

エージェント間でメッセージを交換するためには、エージェントの実行環境やタスクに関する記述など知識フレームワークを共有することができるエージェントコミュニケーション言語が必要である。1つの分散環境におけるエージェント間通信を行なうためにこの KQML を用い、分散環境にあるオブジェクトのクライアント/サーバ通信は、CORBA により連動させる。KQML は、ask-all や tell といった 24 種類の予約済み Performative により操作を記述する。メッセージ内容表現形式 KIF (Knowledge Interchange Format)、属性から構成され、お互いのエージェントが交換するメッセージの存在 (ontology) を共有するために、コミュニケーションファシリテータ (AWFE) を通じて行なう。CORBA は、オープンかつマルチベンダなアーキテクチャであり、インタフェース部分を IDL (Interface Definition Language) により記述し、オブジェクトのカプセル化を行っている。

2.6 XML

現在、アプリケーション間で XML ドキュメントを交換するプロトコルとして SOAP (Simple Object Access Protocol) がある。下位のトランスポートプロトコルは、一般的に HTTP が利用されている。CORBA 環境で Web サービスの実装基盤として利用するためには、SOAP と IIOP のインタオペラビリティが必要であり、SCOAP (Simple CORBA Object Access Protocol) プロトコルの標準化が進められている。

XML をエンドユーザインタフェースとする場合、インタフェース構造の変更や視覚的効果が期待できる反面、IIOP ではなく SOAP で送るとメッセージサイズが膨張し、ネットワークトラフィックの増大に繋がる。また、XML パーサの負荷による遅延も想定される。そのため、今回は、IDL を用いる。また、実装言語には、Java を用いてエンドユーザとの

インタフェースには、Java-XML変換を行い、GUI環境を提供するものとする。

3. システム実装と実際例

3.1 実験環境

今回のシステムは、販促企業と数社の製造企業からなるシステムを考える。図 3.1 が示すように、各企業の通信は、インターネットを介して XML で行われ、製造企業には、各社の CAD データベースが存在し、販促企業には、全体の工程を管理するための Web/ワークフローサーバが存在する。

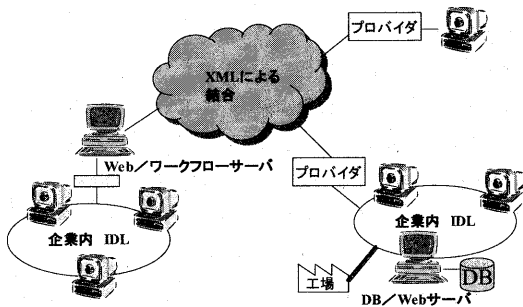


図 3.1 システム概観
Fig.3.1 System overview.

インターネットへの接続は、専用回線、ダイアルアップいずれでも有り得るが、Web/ワークフローサーバには、グローバル IP を設定する必要がある。さらに、Apache のアクセス制限と SSL 機能を用いることで、グループ企業のユーザのみがワークフローページにアクセス可能となる。

3.2 ワークフロー

発注から納品までの企業活動を業務プロセスの集合であると捉えれば、今回のシステムは、図 3.2 に示すような業務プロセス図となる。このような複数の企業間における業務プロセスを自動化し管理するためにワークフローを用いる。

さらに、ハイブリッド型発電機製造過程において、ヒアリングを行い、実際の業務プロセスを具体的に表現したものがプロセス定義となる。

る。(図 3.3)

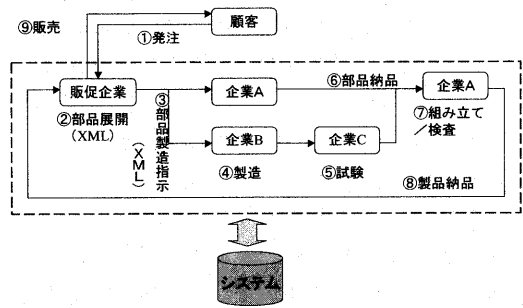


図 3.2 業務プロセス
Fig.3.2 Business process.

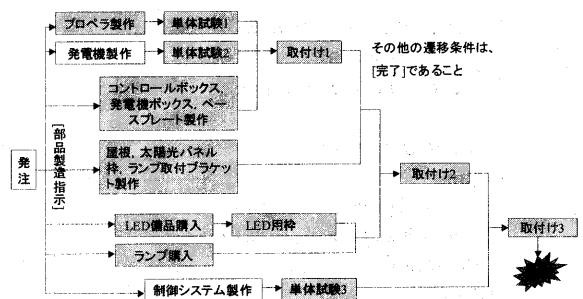


図 3.3 プロセス定義
Fig.3.3 Process definition.

3.3 データベース

実装されるデータベースは、CAD データ管理、顧客データ管理、ワークフロー制御の 3 つを目的として搭載される。CAD データと部品データの関係は、n:n となり得るため、リンキングテーブルを経由して 1:n でリレーションシップを構築する。

また、製造途中に新たな発注が発生する場合があります。1つのプロセスに対して複数のジョブが発生することを考慮してワークフロー制御を行なうことが必要である。プロセス間の連結は、工程連結テーブルに記述され、ECA テーブルを参照しながら、不可逆的にプロセスが進行する。ジョブの管理には、ID を用い、プロセスの完了は、共通エリアにある ECA 制御テーブルに記述されるものとする。従って、一連の製造が終了した時点で、スーパーバイザプログラムが、ECA 制御テーブルのクリアを行なう。

(図 3.5)

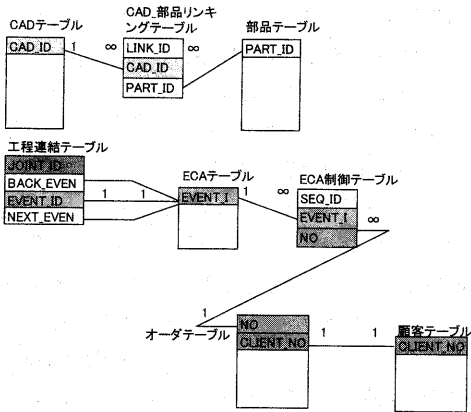


図 3.4 リレーション
Fig.3.4 Relations.

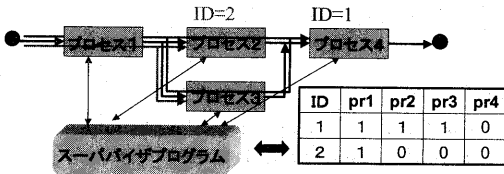


図 3.5 ID によるプロセス制御
Fig.3.5 Process control.

3.4 ECA ルール

イベント（環境内に起こった事象）、コンディション（ルールを実行するための制約条件）、アクション（実際に行なう動作）の3つを一組にして動作の記述を行なうことをECAルールという。更新処理が発生したときに、商品数が5個以下であれば、10個になるように発注を行なうというプロセスにおいては、イベント「更新処理発生」、コンディションは、「商品数が5個以下である」、アクションは、「10個になるように発注を行なう」となり、図3.6のような表記となる。

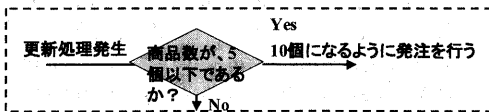


図 3.6 ECA ルール
Fig.3.6 ECA rule.

ここで、アクション（メソッド）について考

える。提示したシステムには、コンディションはないので、個々の業務プロセス完了で次へ移行している。イベントとアクションの関係を表3.1に示す。

表 3.1 アクティブルール
Fig3.1 Active rules.

Trigger

プロペラ製作	単体試験1
発電機製作	単体試験2
単体試験1	取付け1
単体試験2	取付け1
コントロールほか製作	取付け1
屋根ほか製作	取付け1
LED備品購入	LED用枠製作
LED用枠製作	取付け2
ランプ購入	取付け2
制御システム製作	単体試験3
取付け2	取付け3
単体試験3	取付け3

Order

プロペラ製作	単体試験1
発電機製作	単体試験2
単体試験1,単体試験2,コントロールほか製作	取付け1
LED備品購入	LED用枠製作
コントロールほか製作,取付け1,屋根ほか製作,LED用枠製作,ランプ購入	取付け2
制御システム製作	単体試験3
取付け2,単体試験3	取付け3

Real-Time

発注	プロペラ製作
発注	発電機製作
発注	コントロールほか製作
発注	屋根ほか製作
発注	LED備品購入
発注	ランプ購入
発注	制御システム製作

3.5 インタフェース

ユーザとのインタフェースには、ブラウザを用い、XML文書にてやり取りを行なう。

- (1) 見積もり書作成
ブラウザから必要な部品を選択することで見積もりを作成し、XMLデータに変換する。
- (2) 部品データ登録/データ展開
・ 1つの部品を製作するために必要な

CADデータの登録を行なう。

- ・ 指示された部品からそれを構成するCADデータを表示させる。

(3) CADデータ登録/検索

- ・ CADデータをDBに登録する際にキーワードも設定する
- ・ ファイル名だけではなく、キーワードからもCADデータを検索する。

(4) XMLビューア

ワークフローの進捗状況をモニタリングして表示し、複数企業間の工程管理を可能にする。

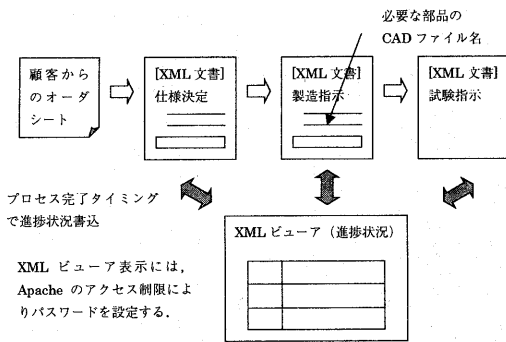


図 3.7 XMLビューア利用イメージ
Fig.3.7 XML viewer.

4 むすび

CADデータを含む発注-設計-製造-納品という分散ワークフローシステムのアーキテクチャとプロトタイプについて論じた。ワークフローのスケジューラとモニターに相当するワークフローエンジンは、ECAルール機構を備えたエージェントベースシステムとして設計した。我々は、既に、オープンでダイナミックな分散環境における異種情報源の統合利用のための、エージェントベースのメデイエーション・システム HI-AMS を提案している¹⁾²⁾。これは、能動データベースとエージェントベースを融合したアーキテクチャであり、今回の研究テーマは、この HI-AMS アーキテクチャを、分散ワークフローシステムに適用し、実際の企業間のワークフロー管理システムとして実装

することを試みた。

関連研究では、ワークフロー管理のデータベースアプローチである WIDE プロジェクトがあり、そこでは、CORBA、トランザクションやECAルール等の要素を取り入れたワークフローシステムの研究がされている³⁾。また、Web上でのCORBAとXMLの要素を取り入れたワークフローベースの研究がある⁴⁾。我々の研究は、上記の関連研究のアーキテクチャを、エージェントベースとデータベースを融合した⁵⁾視点からアプローチした点の特徴である。

今後の課題は、プロトタイプシステムに実際の企業情報を取り入れて、システムの有用性を検証することである。また、HI-AMSで提案している拡張トランザクションのアルゴリズムを実験して行くことである。

参考文献：

- 1) Konishi,O., "Design and Implementaion of Active Mediation System -A study on Agent Based Databases-", In Advanced Database Systems for Integration of Media and User Environments'98, World Scientific, pp.149-152.
- 2) 小西 修, 木内 光, "分散協調処理におけるエージェントベース・トランザクション", 電子情報通信学会信学技報, Vol.98, No.486, DE98-36, PP.49-56, 1998年12月.
- 3) P.Grafen, B.Pernici, and G.Sanchez, "Database Support for Workflow Management -The WIDE Project", KLUWER ACADEMIC PUBLISHERS (1999).
- 4) A.Dogac, et al., "A Workflow-based Electronic Marketplace on the Web," SIGMOD Records, 1998.
- 5) E.Pitoua and B.Bhargava, "A Frame-work Providing Consistent and Recoverable Agent-Based Access to Heterogeneous Mobile Databases," SIGMOD Records, Vol.24, No.3, pp.44-49, Sept.1995.