2005－OS －98（14）
2005／2／23

ユビキタス環境におけるトランザクション処理方式

西山 正浩 †　毛利 公一 ††　大久保英嗣 ††

† 立命館大学大学院理工学研究科　†† 立命館大学情報理工学部

現在, 我々は, ユビキタス環境におけるトランザクション処理のためのミドルウェアについて研究している．本ミドルウェアでは, トランザクション処理に関する一貫性に関して, 従来の強い一貫性のみならず, 弱い一貫性を実現するために, 入れ子型トランザクションを採用している．また, ユビキタス環境における様々なサービスの実現を容易にするために, イベント駆動型のトランザクションやデータのモニタリングのためのトランザクションを始めとして, 様々な種類のトランザクションの検討も行っている．さらに, 利用者, データ, トランザクションの位置と時間を始めとした環境情報（コンテキスト）を使用することによって, 様々なサービスを提供可能とする枠組みの検討も行っている．本稿では, このようなユビキタス環境におけるトランザクション処理の検討項目とシステムの基本構成について述べる．

A Transaction Processing Scheme in Ubiquitous Computing Environment

Masahiro Nishiyama †　Koichi Mouri ††　Eiji Okubo ††

†Graduate School of Science and Engineering, Ritsumeikan University
††College of Information Science and Engineering, Ritsumeikan University

We have been studying a middleware for transaction processing in ubiquitous computing environment. This middleware adopts the nested transaction in order to support not only the conventional robust consistency but also the concept of weak consistency. Furthermore, in this study, in order to facilitate implementation of various services in ubiquitous computing environment, we have been examining a variety of kinds of transactions such as the event-driven transaction and the data-monitoring transaction. We also have been examining the framework that can provide various services by utilizing environmental information (context) including the location information and time information of users, data, and transactions. In this paper, design issues and the basic system configuration for tansaction processing in ubiquitous computing environment are described.

1　はじめに

近年，ハードウェア技術の発展に伴い，計算機の小型化，高性能化が進み，携帯電話やPDAといった小型計算機が急速に普及している．例えば，携帯電話は，カメラやGPS機能に止まらず，ICタグ機能など，さまざまな機能を取り入れ，今後ますます高機能化すると予想される．また，ネットワーク機器の整備により，小型計算機の端末から，ネットワーク上に遍在する資源やサービスを利用可能とする環境が整いつつある．以上のことから，このような環境上に遍在する機器や利用者の持つ端末から形成されるユビキタスコンピューティング環境の実現や，サービスのためのソフトウェアプラットフォームの実現が望まれている．

ユビキタス環境とは，身近なあらゆる機器にコンピュータが組み込まれ，利用者がそれらを意識することなく，それらの機器や機器が提供するサービスを利用可能とする環境である [1]．このような環境において，利用者は，偏在する機器やサービスにさまざまな要求を行うと考えられる．このため，利用者の求めるサービスを実現するには，単に機器やサービスを利用できるだけでなく，機器と機器，機器とサービスとを組み合わせて新たなサービスを創出するサービスの合成，連携技術が重要であると考えられる．

以上の背景から，現在，我々は，ユビキタス環境に適したトランザクション処理方式を研究している．本研究では，トランザクション処理の概念をユビキタス環境へ適応することで，サービスの合成，連携技術を確立し，サービスのソフトウェアプラットフォームを実現することを目的としている．

トランザクションは，BEGINとENDで囲まれた一連のデータ操作であり，データベースやオンライン商取引などの分散処理の信頼性を向上させるための概念として発展してきた [2]．一方，ユビキタス環境では，ネットワーク上の資源を利用するという分散環境の側面と，利用者が携帯端末を持ち移動する

表 1　ユビキタス環境のコンテキスト

コンテキスト	詳細
位置	利用者，データ，サービス
時間	過去，現在，未来
環境	計算機，ネットワーク，センサ
好み	利用者，履歴

というモバイル環境 [3] の側面がある．これらの環境上のシステムは，それぞれ，位置透過性（location transparency），位置適応性（location awareness）を目的に実現されている．このため，ユビキタス環境上のシステムは，これらの目的を包含するコンテキスト適応性（context awareness）を目的に実現されると考えられる．ユビキタス環境において我々が想定しているコンテキストを表 1 に示す．

以下，本稿では，2 章でトランザクション処理における検討事項について述べ，3 章でシステムの構成について述べる．さらに，4 章でトランザクション処理方式について述べる．

2　トランザクション処理における検討事項

トランザクション処理は，分散処理を信頼できるものとするソフトウェア技術で，通信，財務，旅行，製造，プロセス制御などの分野で発展してきた．また，トランザクション処理システムは，分散コンピューティングやフォールトトレラントコンピューティングの分野において，多くの概念を創出してきた．クライアントサーバモデルやリモートプロシジャコールは，その代表である．さらに，トランザクション処理は，分散処理のための統一的概念として，ACID 特性，すなわち，原子性（atomicity），一貫性（consistency），分離性（isolation），持続性（durability）という概念を導き出した．

ユビキタス環境のトランザクション処理方式は，トランザクションの一貫性を弱め，いくつかのトランザクションモデルを採用することで実現する．本章では，これらに関する検討事項について述べる．さらに，トランザクション処理方式がコンテキスト適応性を実現するための位置管理についても述べる．

2.1　システムの想定環境

本研究は，ユビキタス環境におけるトランザクション処理方式を明らかにし，サービスのソフトウェアプラットフォームを実現することを目的としている．本システムの想定環境を図 1 に示す．以下に，各構成要素の役割を説明する．

ディレクトリサービス サービスの管理，サービスの利用者管理の役割がある．

サービスゲートウェイ サービス利用のための窓口としての役割がある．

サービスプロバイダ サービスゲートウェイからの要求でサービスを提供する．他のサービスプロバイダ宛の要求を転送することもできる．

図 1 では，利用者が携帯端末を持ち，無線通信を使用してサービスを要求したのち，処理結果を待たず移動している様子を表している．この場合，利用者の端末がサービスゲートウェイの電波到達範囲外へ移動したため，ネットワークへの接続性が切れ，別のネットワークへ再接続し，別のサービスゲートウェイから処理結果の受取りを要求している．

利用者は，ディレクトリサービスを利用してサービスの検索を行い，サービスを発見する．複数のサービスを連携させることもできる．また，利用者は，ディレクトリサービスを利用することで，サービスゲートウェイのアドレスを知ることができる．

利用者からの要求は，サービスゲートウェイが受け取る．サービスゲートウェイは，利用者のサービスの利用権限があるか否かをディレクトリサービスに問い合わせ，権限がある場合には，要求されたサービスをトランザクションとして発行する．この際，サービスゲートウェイは，トランザクションに一意の識別子を生成する．利用者の端末は，この識別子を記憶し，処理結果の受け取りの際に使用する．サービスゲートウェイは，発行したサービスのサービスプロバイダへ処理を依頼する．サービスプロバイダは，サービスの処理結果を自身の持つメッセージボックスに書き込み，次のサービスを実行するため，サービスプロバイダに処理を依頼する．利用者は，移動後，要求時のサービスゲートウェイのアドレスとトランザクションの識別子を指定して，処理結果を受け取ることができる．

2.2　トランザクションモデル

ユビキタス環境におけるサービス間の複雑な関係を実現するために，以下に述べる複数のトランザクションモデルを適用する．サービス間がどのトランザクションモデルを採用するかは，サービス要求時にサービスゲートウェイへ送信されるメッセージに記述することができる．

（1）フラットトランザクション

フラットトランザクションは，最も単純なトランザクションモデルである．フラットトランザ

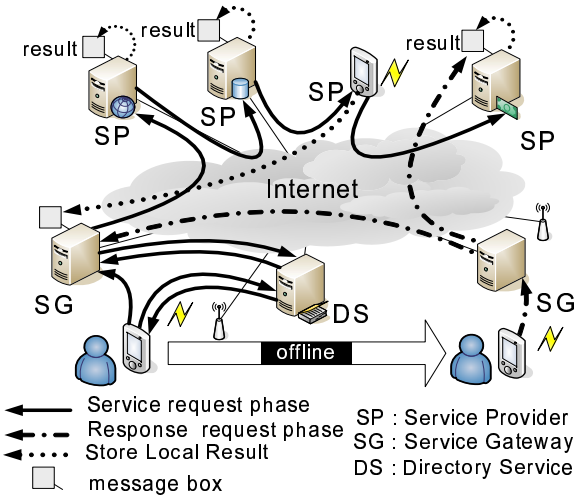

図 1　想定環境

クションは，アプリケーションを原子的な動作に組み立てるための基本構成要素である．フラットトランザクションは，制御のための階層が 1 つしかなく，そのトランザクションは，コミットで他のすべてのものと一緒に生き残るか，アボートで他のすべてのものと一緒にロールバックされるかのどちらかである．

本システムでは，サービス間で強い一貫性（robust consistency）が求められる場合にフラットトランザクションを採用する．

（2）連鎖型トランザクション

連鎖型トランザクションは，原子的動作のシーケンスを一度に 1 つ実行させるトランザクションモデルである．連鎖中のトランザクションでは，あるトランザクションのコミットによって，別のトランザクションが開始される．

本システムでは，サービス間で連続的な関係が求められる場合に連鎖型トランザクションを採用する．

（3）入れ子型トランザクション

入れ子型トランザクションは，トランザクションの階層構造を形成可能とするトランザクションモデルである．入れ子型トランザクションでは，呼出し関係に基づく親子関係として表現される．入れ子型トランザクションでは，同一階層のサブトランザクションを並行起動し，トランザクションの効率を高めることができる．サブトランザクションは，個々にコミットかロールバックのいずれかができるが，子のコミットは親のコミットまで効力を発揮しない．また，あるトランザクションのアボートは，それに属すすべてのサブトランザクションだけがロールバックされる．

本システムでは，サービス間で階層的な関係が求められる場合に入れ子型トランザクションを採用する．

（4）イベント駆動型トランザクション

イベント駆動型トランザクションは，あらかじめ記述したルールに適合するイベントが発生した際に指定したトランザクションを発行するトランザクションモデルである．ルールには，周期的，非周期的イベントを記述できる．

本システムでは，サービスが特定のイベントを契機にトランザクションの発行を必要とする場合にイベント駆動型トランザクションを採用する．

（5）長時間トランザクション

長時間トランザクションは，従来の方式と比べてトランザクションを長時間実行可能とするトランザクションモデルである．長時間トランザクションでは，トランザクション間の分離性の制約を緩和することで，他のトランザクションの終了まで資源をロックすることがなく，スケーラビリティを向上できる．

（a）モニタリングトランザクション

モニタリングトランザクションは，指定された値の変化を監視し続け，値の変化を契機にトランザクションを実行するトランザクションモデルである．

本システムでは，サービスがセンサなど特定の値の変化を契機にトランザクションの発行を必要とする場合にモニタリングトランザクションを採用する．

（b）モバイルトランザクション

モバイルトランザクション [4] は，1 台以上のモバイル端末があるモバイル環境で，モバイル端末のネットワークへの接続性や位置の変化に適応したトランザクションモデルである．モバイルトランザクションでは，トランザクション処理の実行，中断，再開を実現する

本システムでは，サービスプロバイダの移動に伴い，サービスが移動する場合にモバイルトランザクションを採用する．

2.3　弱い一貫性

入れ子型トランザクションでは，同一階層のサブトランザクションは，それぞれ一貫性を持ち，並行

起動される．このため，いずれかのサブトランザクションが失敗した場合には，親トランザクションの実行や同一階層のサブトランザクションのコミットが中断される．

しかし，ユビキタス環境では，サブトランザクションのいずれかがコミットできればよい状況などが考えられ，従来型の強い一貫性よりも， 弱い一貫性（weak consistency）[5] の方が適している場合がある．これらの状況に適応するため，本方式では，以下に示す 4 種類の実行モードを設けている．

（1）通常モード

通常モードは，強い一貫性を必要とするトランザクションの実行に用いるモードである．このため，このモードのトランザクションは，強い一貫性の入れ子型トランザクションの手順でコミット処理が行われる．サブトランザクションが破棄された場合，同一階層のサブトランザクションも破棄され親トランザクションのアボート処理が行われる．

（2）先行コミットモード

先行コミットモードは，サブトランザクションのコミット処理における制限を緩和したモードであり，サブトランザクションは，親トランザクションの終了を待つことなく先行して完了することができる．このモードで実行するためには，当該サブトランザクションの実行を打ち消す作用を持つ補償トランザクションが定義可能であることが条件となる．先行コミットモードのサブトランザクションが完了したのち，親トランザクションがアボートした場合には，システムが自動的に補償トランザクションを実行することでその実行を打ち消す．

（3）単独破棄モード

単独破棄モードは，サブトランザクションが破棄されても親トランザクションがアボートされないモードである．このため，アボートしたサブトランザクションと同一階層のサブトランザクションも破棄されない．

（4）独立モード

独立モードは，コミットおよび破棄における制限を両者とも緩和したモードである．このため，このモードで実行されるトランザクションは，コミットおよび破棄のいずれの状態で終了しても他のトランザクションに影響を与えることはない．このモードでは，利用可能条件および利点ともに，先行コミットおよび単独破棄モードの内容を持つ．

2.4 位置管理

本研究では，トランザクション，サービス，利用者の位置の変化を許容している．

- トランザクションの位置
  トランザクションは，サービスプロバイダを移動しながら実行される．どこのサービスプロバイダで実行されているかという情報は，サービスゲートウェイが管理する．この情報を管理することで，利用者は，実行途中のトランザクションを強制的にアボートしたり，途中結果を要求することができる．
- サービスの位置
  サービスプロバイダの位置は，ディレクトリサービスによって URI として管理されている．サービスプロバイダは，移動前，移動中のサービスの無効化を依頼する．移動後，ディレクトリサービスへ移動先の URI を伝えることで，再びサービスを提供できる．
- 利用者の位置
  本システムでは，利用者はサービス要求時を除いて，利用者の移動によるサービスの中断は発生しない．利用者の位置は，センサやリファレンスタグなどで得ることができる．このため，本システムでは，これらのデバイスとトランザクションの連携によって，位置依存型サービスを実現できる．

2.5 耐障害性

障害発生時，サービスプロバイダのメッセージボックスは，ログとしての働きを担う．すなわち，メッセージボックスに保存されている処理結果は，障害直前に行われたトランザクションの処理結果であり，そのデータからトランザクション処理を復旧できる．以下に，サービスプロバイダ，サービスゲートウェイ，ディレクトリサービスの耐障害性を述べる．

サービスプロバイダ

サービスプロバイダにおける障害発生時のトランザクション処理は，直前のトランザクション処理の結果をメッセージボックスから取り出し，登録された代替サービスプロバイダで処理を行うことで復旧できる．

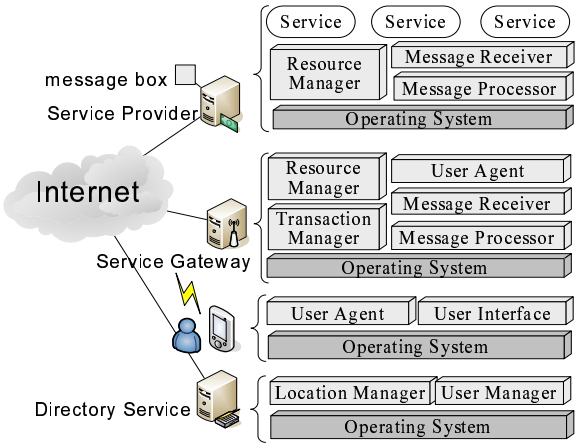

図 2　システムの全体構成

サービスゲートウェイ
サービスゲートウェイにおける障害発生時は，自身の代替サービスゲートウェイをディレクトリサービスに登録し，サービス要求の宛先を変更することで対応する．

ディレクトリサービス
ディレクトリサービスにおける障害発生時は，冗長関係にある他のディレクトリサービスが，自動的にディレクトリサービスを担う．

3　システム構成

本システムの全体構成を図 2 に示す．本システムは，オペレーティングシステム上のミドルウェアとして実装している．システムは，Java で記述しており，サービスの要求には SOAP[6] を用い，その通信プロトコルには HTTP または SMTP を用いる．本システムは，以下の要素から構成される．

トランザクションマネージャ
トランザクションマネージャは，2.2 節で述べたすべてのトランザクションモデルのコミットメント制御と，一貫性の管理を行う．本システムでは，弱い一貫性を採用しており，それらに沿った一貫性の管理も行う．

リソースマネージャ
リソースマネージャは，サービスの成否を監視する機能と，トランザクションマネージャにコミット/アボートを依頼する機能がある．

メッセージレシーバ
メッセージレシーバは，従来の HTTP サーバ，SMTP サーバの機能を持ち，SOAP メッセージを受け取ると，メッセージプロセッサにメッセージを渡し解析処理を依頼する．

メッセージプロセッサ
メッセージプロセッサは，SOAP メッセージの解析，返答メッセージの作成のための機能がある．

ユーザインタフェース
ユーザインタフェースは，利用者の端末に存在し，利用者がサービスを要求するための仲介の役割を果たす．

ユーザエージェント
ユーザエージェントは，利用者の端末に存在し，利用者がサービスを供給するための仲介をなす．

ディレクトリサービス
ディレクトリサービスは，サービスの位置を管理し，サービスを利用するために，利用者，サービスの位置を管理する．また，ディレクトリサービスは，サービスゲートウェイのアドレスも保持している．ディレクトリサービスは，ロケーションマネージャとユーザマネージャの機能を持つ．

ロケーションマネージャ
ロケーションマネージャは，利用者とサービスの位置を管理する．利用者の位置は，センサやリファレンスタグからの情報で特定する．サービスの位置は，サービスの登録時に指定される．また，サービスの位置は，携帯端末の移動などで，変更することも可能とする．また，ロケーションマネージャは，サービスゲートウェイのアドレスも管理している．

ユーザマネージャ
ユーザマネージャは，サービスを利用可能なユーザ，会社，地域，仲間を管理する．このため，サービスは，特定の仲間のみが利用できるなど制限を加えることができる．

メッセージボックス
メッセージボックスは，固定ホストのサービスプロバイダおよびサービスゲートウェイ上にあり，サービスの処理結果を格納する．
サービスの処理結果を自身でも持つことで，障害時にはログとしての役目を果たす．

4　トランザクション処理方式

4.1　データアクセス方式

本システムの特徴的なデータアクセス方式には，集中処理方式（図　3 参照）と巡回処理方式（図 4

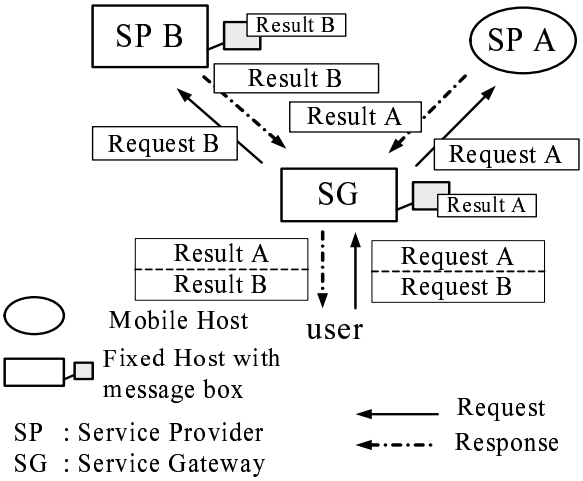

図 3　集中処理方式

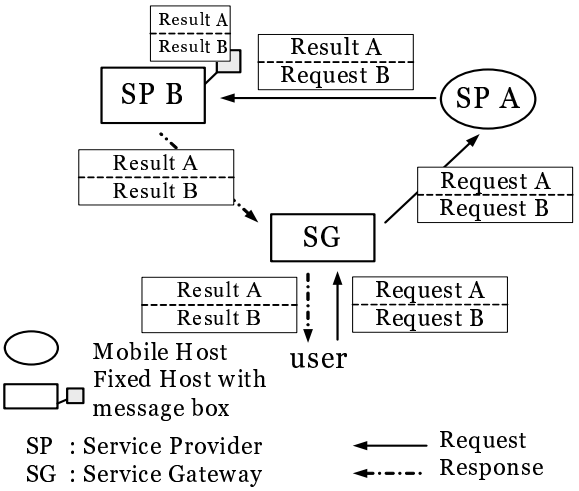

図 4　巡回処理方式

参照）がある．

集中処理方式では，利用者からサービス要求を受け取ったサービスゲートウェイが要求されたサービスプロバイダへ処理を要求する．この場合，要求されたサービスは，すべてのデータが揃うまで待ち，要求元へ一括して結果を返す．

巡回処理方式は，サービスプロバイダによるサービスの転送によって実現される．巡回処理方式の適用場面としては，データベースの巡回探索が考えられる．

5　おわりに

本稿では，ユビキタス環境におけるトランザクション処理方式について述べた．ユビキタス環境のトランザクション間には，複雑な関係があると考えられる．本方式では，複数のトランザクションモデル用いて複雑な関係を記述できる．これにより，位置，時間，環境の情報などとトランザクションとを連携させることができ，トランザクションがコンテキストに適応可能となる．

また，本方式に基づくサービス連携のためのソフトウェアプラットフォームの構成について述べた．本システムでは，サービスの処理結果をサービスプロバイダ自身が持つメッセージボックスに残す．これにより，利用者は，サービス要求後，処理結果を待たず移動することができる．

今後は，利用者の移動に適応するメッセージボックスの移送手法やを検討する予定である．

参考文献

[1] Weiser, M.: The Computer for the 21st Century, *Scientific American*, Vol. 265, No. 3, pp. 94–104 (1991).

[2] ジム・グレイ, アンドレアス・ロイター：トランザクション処理 概念と技法 (上), 喜連川優監訳, 日経 BP 社 (2001).

[3] Barbará, D.: Mobile Computing and Databases - A Survey, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 11, No. 1, pp. 108–117 (1999).

[4] Serrano-Alvarado, P., Roncancio, C. and Adiba, M.: A Survey of Mobile Transactions, *Distributed Parallel Databases*, Vol. 16, No. 2, pp. 193–230 (2004).

[5] 國枝和雄, 各務達人, 大久保英嗣, 津田孝夫：分散トランザクションシステム IXI の設計と実装, 情報処理学会論文誌, Vol. 35, No. 6, pp. 1185–1199 (1994).

[6] SOAP: Simple Object Access Protocol, http://www.w3.org/TR/soap/.

[7] Cao, J., Feng, X., Lu, J. and Das, S. K.: Mailbox-Based Scheme for Mobile Agent Communications, *IEEE Computer*, Vol. 35, No. 9, pp. 54–60 (2002).

[8] Dunham, M. H., Helal, A. and Balakrishnan, S.: A Mobile Transaction Model That Captures Both the Data and Movement Behavior., *Mobile Networks and Applications*, Vol. 2, No. 2, pp. 149–162 (1997).

[9] 永田智大, 西尾信彦, 徳田英幸：サービス利用状況の変化に対する適応支援機構, 情報処理学会論文誌, Vol. 44, No. 3, pp. 835–847 (2003).