モバイル環境における端末とサーバ間のデータ同期方式

1B－8

入宮 貞一　箱守 聰　井上 潮
NTTデータ通信 情報科学研究所

1 はじめに

最近のネットワーク基盤の整備や計算機の高性能化などにより、電子化された各種の情報をネットワーク経由で利用できるようになっている。このような中で、いつでもどこでも必要なデータにアクセスできる環境が望まれている。しかし、携帯端末と無線ネットワークから構成されるモバイルコンピューティング環境(以下、モバイル環境)は接続が不安定で、接続コストが固定ネットワークと比べて高い。そのため、ネットワークへ接続できない場合にもデータへのアクセスが可能な、コストの低いデータ管理方式が求められている。

本稿では、モバイル環境において、携帯端末とサーバの双方でデータ更新が生じる場合のデータ管理方式について検討し、要求条件、課題、対処および処理方式について述べる。

2 環境モデル

前提とする環境モデルを図1に示す。環境は固定ネットワークと複数のモバイルネットワークから形成されている。固定ネットワークにはWWWサイトやデータベースサイトが接続されている。モバイルネットワークには複数の携帯端末(以下、端末)が存在する。

このような環境において、本人(所有者)からの参照/更新要求と他者(所有者以外)からの参照/更新要求が生じる処理を考える。モバイルネットワークは接続が不安定であり、端末はネットワークに接続できないことがある。その場合でもデータへのアクセスを可能とするために、データの原本を固定ネットワーク上のサーバに置き、原本の一部(キャッシュ)を端末に置く。そして本人からのアクセスは端末で、他者からのアクセスはサーバで受け付ける。また、端末がネットワークと接続した時点で、端末とサーバ間でデータの整合をとる。

このように、データの原本を固定ネットワーク上のサーバに置くことは、端末の盗難や落下などの事故に対

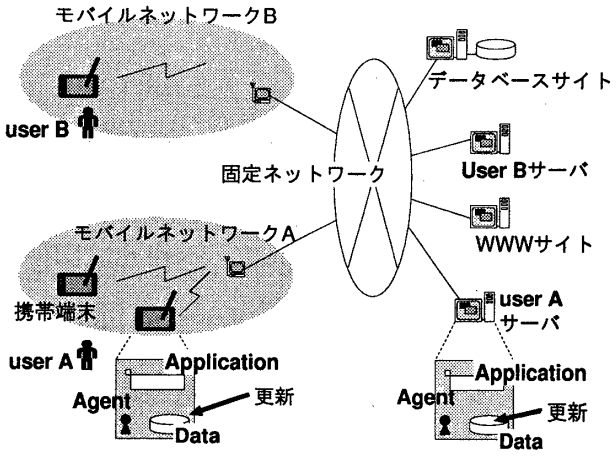

図1: 環境モデル

するデータの安全性の向上にも役立つ。

3 課題

2章で述べた処理を実現する場合、以下の点を考慮する必要がある。

要求1 端末への要求とサーバへの要求は要求者が異なっているため、それぞれの役割に応じた処理をする必要がある。

要求2 実際のアプリケーションの中では、ファイルの一部分を更新するものが多く、ファイルの一部分の更新に適したデータ管理が必要である。

モバイル環境におけるデータ管理を対象とした従来の研究には、Coda[1]とBayou[2]があるが、Codaは個々のファイル単位でデータを管理しており、Bayouは要求者の区別がないため、そのまま利用することはできない。以上のことより、次の点が課題となる。

課題1 要求1と要求2を考慮した、ネットワークの切断状態でのデータアクセス方式。

課題2 要求1と要求2を考慮した、端末とサーバで同一データを同時更新したときのデータの不整合の検出とその解決方法。

A Data Synchronization Mechanism in Mobile Computing
Sadaichi IRIMIYA, Satoshi HAKOMORI and Ushio INOUE,
Laboratory for Information Technology, NTTDATA Co.,
Email:{sada,hako,uinoue}@lit.rd.nttdata.jp

4 提案方式

要求1に対応するためには、本人からの更新要求は端末とサーバで同期をとって更新すること、他者からの更新要求は本人の確認を経て更新すること、の2つの処理が必要となる。これらの要求を満たし課題を解決するために以下の方式を提案する。

まず、端末とサーバにデータの更新要求を保存するスプールを用意する。そして、更新要求が来ると、すぐには原本やキャッシュを更新せず、全てスプールに一時的に保存する。一方、参照要求に対しては、スプール中のデータと原本やキャッシュ中のデータを管理プログラムで統合し、仮想的に単一のデータとして提示する。

以上の対処により、3章の課題は次のように解決できる。まず、課題1については、参照や更新の要求を端末とサーバのスプールで処理する事により、ネットワークが切断されている状態でのデータアクセスが可能となる。また、課題2に関しては端末とサーバのスプールを比較することで不整合の検出が可能となる。このとき、原本やキャッシュ中のデータを直接比較する必要がないため、効率的な処理が期待できる。

スプールを用いた参照や更新処理は、具体的に以下のように行う。

(1) ネットワーク切断時: 参照/更新要求はスプール上で保存する。

(2) ネットワーク接続時: ネットワークに接続した時点で、端末とサーバ間のスプール中の要求の整合性を検査する。不整合状態と判断するのは、同一データに対して、双方から更新が要求されている場合と、一方から参照が、他方から更新が要求されている場合である。これ以外の場合には整合状態とし、要求を原本とキャッシュに反映する。不整合状態では、その旨を要求者へ通知し、原本とキャッシュへの反映は行わない。

5 更新要求時の処理

4章で述べた提案方式を用いて更新要求を処理する時の流れを図2に示す。以下、本人から要求された場合と他者から要求された場合に分けて説明する。

- 本人からの更新要求

(1) 要求を端末のスプールに保存する。

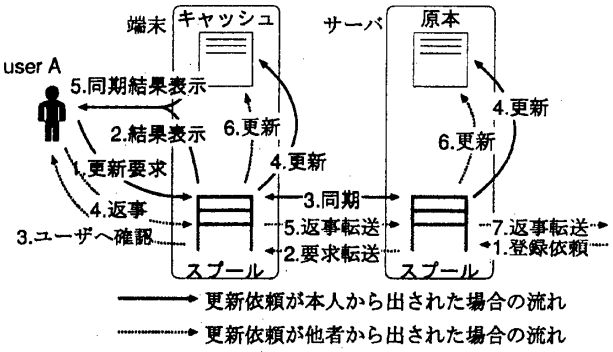

図2: 更新要求の処理の流れ

(2) 端末とサーバの接続を待ち要求をサーバへスプールする。

(3) 端末とサーバで同期をとって原本とキャッシュを更新する。

(4) 結果を本人へ通知する。

- 他者からの更新要求

(1) 要求をサーバにスプールに保存する。

(2) 端末とサーバの接続を待ち要求を端末へスプールする。

(3) 本人に要求の可否を確認してもらい、結果を端末にスプールする。

(4) 端末とサーバの接続を待ち結果をサーバへスプールする。

(5) 本人の判断がOKの場合にのみ、キャッシュの更新を行うと同時に原本を更新する。

(6) 結果を要求者(他者)へ通知する。

6 おわりに

本稿では、モバイル環境において端末とサーバの双方でデータの更新が発生する場合に、スプールを用いたデータ同期方式について提案した。今後は、プロトタイプシステムを作成し、方式の有効性を検証する予定である。

参考文献

[1] Satyanarayanan, et al.:"Disconnected Operation in the Coda File System", ACM Trans. Comp. System., Vol.10, No.1, pp.3–25(1992)

[2] Hauser, et al.:"Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System", Proc. ACM Symp. on Operating System(1995)