オーディオビジュアル複合情報処理　38－6
(2002. 11. 14)

環境適応型オンラインストーレジにおける複製管理方式の評価

中村　元紀†　　井上　知洋†　　久保田　稔†

† NTT 未来ねっと研究所　〒180–8585 東京都武蔵野市緑町 3–9–11
E-mail: †{motonori,inoue,kubota}@ma.onlab.ntt.co.jp

あらまし　将来あらゆる場所にコンピュータが存在し，それらが動的にネットワークを構成して通信しあうユビキタス環境が実現された場合，様々なコンピュータがいつでも共通的にアクセスできるデータはオンラインに保存しておくべきである．一方，それらのコンピュータは無線リンクなどを通じてその場に応じて一時的なネットワークを構成する．このような動的なネットワークトポロジの頻繁な変化はオンラインデータの可用性の低下を招く．筆者らは動的な環境の変化に適応して可用性の低下を防ぐ，分散ストーレジ管理方式を提案してきた．本稿では，提案方式の有効性を定量評価により確認した結果について述べる．

キーワード　分散ストーレジ，ユビキタスコンピューティング，ユビキタスネットワーク，ユビキタスサービス，複製管理，データの一貫性

An Evaluation of Replicas Management Method in Circumstances Adaptive Online Storage System

Motonori NAKAMURA†, Tomohiro INOUE†, and Minoru KUBOTA†

† NTT Network Innovation Laboratories　Midori-Cho 3–9–11, Musashino-Shi, Tokyo, 180–8585 Japan
E-mail: †{motonori,inoue,kubota}@ma.onlab.ntt.co.jp

Abstract　Computers will exist in all places in the future, and when ubiquitous cmputing environment where they constitute opportune networks dynamically is realized, data which can be accessed commonly with anytime should retain in the online. On one hand, dynamic change of network topologies causes the decrease of availability of the online data. We have proposed a distributed storage management system which is adapted for dynamic change of environments and prevents the fall of availability. In this paper, the evaluation result is described, and the effectiveness of the proposed method is explained.

Key words　Distributed Storage, Ubiquitous Computing, Ubiquitous Network, Ubiquitous Service, Replicas Management, Data Consistency

1. まえがき

あらゆる場所にコンピュータが埋め込まれ，それらがネットワークに接続された環境において，ユーザに最適なサービスを提供するユビキタスサービスへの期待が高まっている．そこではユーザが端末を持ち歩き，常にネットワークを経由して情報を獲得する「端末携帯型」のユビキタスサービスと，環境に埋め込まれた機器やセンサが通信を行い，アクチュエータやディスプレイ，音声出力装置などの物理インタフェースを通してユーザにサービスを提供する「環境埋め込み型」のユビキタスサービスが考えられる．

ここで，端末携帯型のユビキタスサービスに関しては，現在携帯電話を持ち歩く生活が普及しているため，PDA を持ち歩く生活にも現実味があるように感じられるが，実際には以下のような問題点が存在する．

- PDA と携帯電話では大きさが異なる．
- 端末を落したり盗まれた時に個人情報を保護するのが難しい．
- PDA を持ち歩くのを忘れると必要な時に必要な情報にアクセスできない．

最初の問題は入力装置も含めて今後のヒューマンインタフェース技術の進展に期待すれば良いが，後者二つの課題はユーザが持ち歩く端末と情報，あるいはサービスが密接に連携しているのが問題の本質である．真のユビキタスサービスでは，ユーザが端末を持ち歩いているかどうかに関わらずサービスを享受できるべきである．

一方端末埋め込み型のユビキタスサービスでは，端末に対して様々な制約や条件が生じる．例えば，大きさを非常に小さくする必要があり，コンピュータの性能が通常の端末に比べて低くなるかもしれない．また，移動する「もの」に埋め込まれる場合を考えると，通信や電源ケーブルの利用が困難であり，無線通信を採用して必要な時のみリンクを張る等の仕組みが必要である．つまり埋め込み型のユビキタスサービスでは，機能の限られた端末同士が協調して最小限の機能や動作でサービスを提供する必要がある．

これらの課題を解決する技術としてオンラインストーレジ(注1)が有効である．オンラインストーレジは，近年のリッチコンテンツの普及，広帯域および常時接続ネットワークの普及，無線帯域の増加によるモバイル環境の充実，などの理由により普及しつつある，ネットワーク内に必要なデータを保存しておいて必要な時にアクセスする技術であり，現在多くのネットワークプロバイダなどが提供している．そこでは，単にデータのバックアップとしての利用だけではなく，様々なアプリケーションをいつでもどこでも同一の環境や設定で起動するために利用したり[1]，効率的な通信の手段としてのデータ保管目的でも利用される[2]．

つまり，オンラインストーレジを用いれば，ユーザ情報やアプリケーションさえもネットワークに保存しておき，必要な時に必要なデータをネットワークから読み出して利用することが可能である．これにより，PDAを忘れても適当な端末上で必要なサービスを享受できる．また，普段持ち歩くPDA上にはユーザの個人情報は保存しないようにし，端末を落したり盗まれた場合にはオンラインストーレジ上のデータをアクセス制限すれば，個人情報が保護できる．更に，オンラインストーレジを介して通信を行うことにより，各端末は必要な時にのみ通信を行ったり起動していれば良い．

一方将来のネットワーク環境を考えると，インターネットや様々な固定ネットワークへ常に接続可能とする方向性だけでなく，ある特定の場所に集まった端末間で一時的に構築されるアドホックネットワークのように，常にあらゆる場所でネットワークを動的に構築する方向の発展も期待されている．それらの一時的なネットワーク自体も固定ネットワークと接続されるかもしれないし，それらとは孤立しているかもしれないし，あるいは接続点を変更しながら固定ネットワークと接続されるかもしれない(いわゆるモバイルネットワーク)．例えば現在でも個人のPDAをデスクトップPCとUSBケーブルで接続してデータの同期をとることがあるが，このPDAはデスクトップPCを経由してインターネットと接続しているわけではなく，PDAとデスクトップPCの間で一時的にローカルなネットワークを構築していると考えることもできる．また，移動する「もの」に埋め込まれた端末間の通信では，その場限りの一時的なネットワークが構築されるのが普通である．

(注1)：一般にはストレージと表記されるが，英語は"storage"であり，カタカナで表記するとストーリッジ，ストーレッジ，ストーレジなどの方が近いため，本稿では「ストーレジ」と表記する．

このような将来のネットワーク環境でオンラインストーレジを利用することを考えると，各端末が常に固定ネットワークと接続しているとは限らない状態で，そのことをユーザやアプリケーションに意識させない技術が必要である．そこで我々は，ユビキタスネットワーク環境で利用するオンラインストーレジとして，環境適応型オンラインストーレジシステムCAOSS(Circumstances Adaptive Online Storage System)を提案している[4]．本稿ではCAOSSの有効性を確認するため，シミュレーションにより評価した結果について述べる．

以下，2.節ではユビキタスサービスで利用されるデータの性質を説明する．3.節では，そのようなデータを利用する際に，ユビキタスネットワーク環境でも可用性の低下を抑えるために提案した複製データ管理方式を説明する．4.節では，提案した方式の有効性をシミュレーション結果で定量的に示す．5.では関連研究について触れる．最後に6.節でまとめと今後の課題を示す．

2. データ管理方式

2.1 データ配置方式

オンラインストーレジのデータ配置方式としては，以下の観点から分類できる．

一括／分割: 一つのストーレジサーバに全てのデータを保存するか，いくつかのデータ毎に異なるサーバに保存するか．一括の場合は負荷集中でアクセスが重くなる，シングルフェイラーポイントになる，などの問題がある．一方分割の場合はそれらの課題を解決できる可能性があるが，分割されたデータへのローケイティング(位置の発見)が難しい．

複製あり／なし: 複製がない場合は，単独データへのアクセスが集中すると上記「一括」の場合と同様の問題が発生する．一方複製がある場合は複製間の一貫性を管理する必要がある．

ユビキタスネットワーク環境では，ネットワークのトポロジが動的に変化するため，「特定のサーバへの接続性」を仮定することはできない．よって，データの分割と複製の生成が必要である．

2.2 複製管理方式

前述のようにユビキタスサービスのためのオンラインストーレジでは複製を持つ必要がある．この時複製間の情報の一貫性を保持する必要があるが，その複製の性質として以下が考えられる．

固定: 複製されたデータが更新されることはないため，一貫性を保持する必要はない．もしデータの内容を更新する場合は別のデータとして新規生成することとなる．

弱整合: 複製されたデータはキャッシュとして扱われる．よってデータ更新時に必ずしも全ての複製にそれが反映されるとは限らず，複製間の一貫性も保証できない．

強整合: 複製されたデータに変更を加える場合，必ず全ての複製データにもその変更を反映して一貫性を保持する．

「固定」の複製は管理が容易だが，更新が不可能なため用途が限定される．また，「弱整合」な複製を実現するには様々な管理方法が考えられる．例えばCoda[5]では，クライアント内

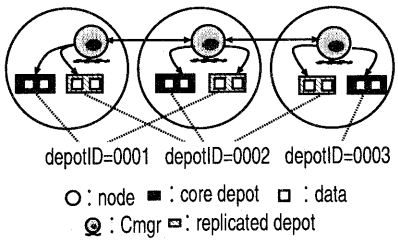

図1 CAOSSの構成要素
Fig. 1 Elements of CAOSS.

にファイルのキャッシュを保持し，サーバと切断された際にはキャッシュに対する更新操作(切断時操作)も許している．このため，切断時操作が発生すると複製間の一貫性は保証されない．よって，データの確からしさが重要な場合には弱整合な複製を利用することはできない．

一方ユビキタスネットワーク環境で複製の数が膨大になった時に「強整合」の複製を利用する場合，データ更新の可用性が低下する可能性がある．つまり，データの利用者と複製の管理を行うサーバが通信不可能になると，データの更新ができなくなってしまう．また，全ての複製にデータの更新を反映するのが難しく，例えばある複製がネットワークから離脱することにより，他の複製全ての更新が不可能となってしまう．

3. CAOSS

前述の問題点を解決して，ユビキタスネットワーク環境での強整合な複製の可用性を高めるため，筆者らは更新操作の可用性の低下を抑えるシステム，CAOSSを提案してきた．なおCAOSSは柔軟な複製管理方式の適用を可能としており，前述の弱整合のデータも扱える[3]が，本稿では説明の簡略化のため強整合のデータ管理方式についてのみ述べる．また4.節においても，強整合の複製管理についてシミュレーション評価を行う．

3.1 構成要素

CAOSSはネットワーク上のサーバやユーザ端末など任意のノードをストーレジとして利用するオンラインストーレジシステムである．以下にCAOSSの構成要素を示す(図1)．

デポ(Depot): 複数のデータを収容した更新の単位．NW内で一意な識別子であるデポIDで識別される．

コアデポ(Core Depot): 更新の権限を持ったデポ．一つのデポIDに対してNW内で唯一存在する．

複製デポ(Replicated Depot): 読み出しのみ可能なデポ．一つのデポIDに対してNW内で0個以上複数存在し得る．

CAOSSマネージャ(CAOSS Manager): デポの操作の受付や分散管理を行う．以下**Cmgr**と表記する．

3.2 管理方式

CAOSSではデポの生成，更新，読み出し処理を以下のように実行する．

- あるノードのCmgrに新規デポの生成が要求されると，そのCmgrは自ノードにコアデポを生成する(図2(a))．デポ生成後そのデポ内に任意のデータを保存することができる．
- コアデポの存在しないノードでそのデポ内のデータに対する読み出し要求を受けたCmgrは，コアデポを管理するCmgrからデータを読み出し，複製デポを生成して要求に応える(図2(b))．ただし，各複製デポには有効期間が設定される．
- コアデポの存在するノードでそのデポ内のデータに対する更新要求を受けたCmgrは，全ての複製デポを管理するCmgrと交渉して更新を実行し，複製にその更新を反映する(図2(c))．この後複製デポからも最新のデータを読み出すことが可能である．
- コアデポの存在しないノードでそのデポ内のデータに対する更新要求を受けたCmgrは，コアデポを管理するCmgrと交渉してコアデポと入れ替えてから更新を行う(図2(d))．なお，ネットワークの状況や要求パタンによっては，入れ替えを行わずに更新を行う場合もある．
- ネットワークトポロジが変化してコアデポと複製デポを管理するCmgr同士が通信できなくなっても，それぞれのデータを読み出すことは可能で，この時データの最新性，すなわち複製デポ間の一貫性は保証できる(図2(e))．ただしコアデポにおいてもデポの更新はできない．
- 通信不可能な複製デポの有効期間が過ぎると，複製デポは無効となる．その後，更新要求が発生しても通信可能な複製デポ間で協調して更新可能となる(図2(f))．

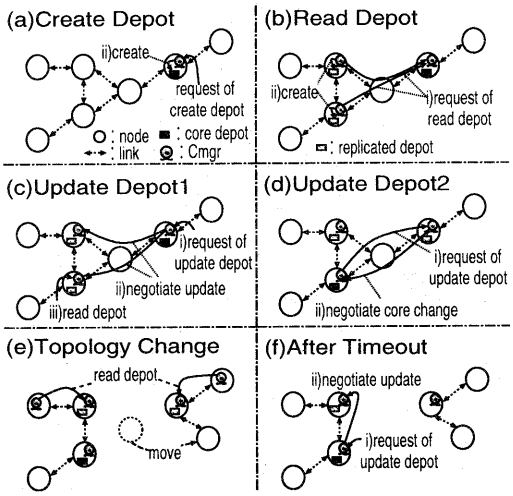

図2 CAOSSの基本動作
Fig. 2 Basic mechanism of CAOSS.

このように，更新要求の発生したノードにコアデポを移動し，複製デポには有効期間を設けることにより，トポロジが変化し易いユビキタスネットワーク環境で，強整合のデータの更新と読み出しの可用性の低下を防ぐことができる．

4. 評価

CAOSSの有効性を示すため，計算機シミュレーションにより評価を行った．本節ではその結果と考察について述べる．

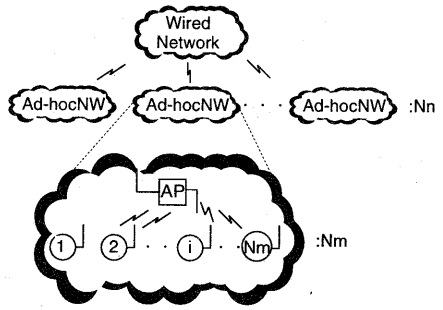

図3 ネットワークモデル
Fig. 3 Network model.

4.1 シミュレーションモデル

4.1.1 ネットワークモデル

ユビキタスネットワーク環境でのCAOSSの有効性を示すため，図3のようなネットワークモデルを仮定した．

- バックボーンである有線ネットワークを通じて N_n 個のアドホックネットワーク (以下**サブネット**) が接続されている．
- 各サブネットには一台のアクセスポイント (以下**AP**) が存在する．このAPは無線I/Fを通じて有線ネットワークと接続するとともに，別の無線I/Fを通じてサブネット内の端末と通信する．有線ネットワークとAPの間のリンクが切れたり再接続することにより，サブネットの移動を模擬する．
- 各サブネット内には (AP以外に) N_m 個の無線I/Fを持った端末が存在する．同一のサブネット内の端末同士は無線I/Fを通じて直接通信可能であるが，異なるサブネットの端末間は各サブネットのAPを経由して通信する．

4.1.2 アクセスモデル

本シミュレーションでは，AP以外の各端末にはユーザが存在し，それぞれストレジアクセスを行うものとする．また，アクセスの種類はデータの読み出し，及び更新の2種類とする．ここで以下のモデルを考える．

- 読み出しはネットワーク内の任意の端末で発生するが，更新は複数の端末で同時には発生しない．
- 要求の発生間隔はポアソン生起に従うこととし，更新の発生したサブネット内での平均間隔を T_i(秒)，他のサブネットでの平均間隔を T_s(秒) とする．ただし $T_i \leqq T_s$ とする．
- 更新の発生した端末では次回も高確率で更新が発生する．更新が連続する確率を P_w とする．

あるデータの更新を行う際に，逐次的に更新内容をオンラインストーレジに保存する利用形態は一般的である．また，サブネットの形態は無線アドホックネットワークを想定しており，物理的にも近い端末の集まりと考えられるため，サブネット間に比べて同一サブネット内でのデータ読み出し頻度が高いのも一般的である．よってこのようなアクセスモデルは現実の利用シーンに近いと考えられる．

4.1.3 複製管理モデル

本稿では，CAOSSにおけるコアデポの移動の有効性を示すため，以下の二つの管理モデルを考える．

表1 シミュレーションパラメタ
Table 1 Simulation parameters.

伝送速度	11Mbps
APでの処理遅延+AP間伝送遅延	1(ミリ秒)
データパケット長	1500(Byte)
制御パケット長	64(Byte)
N_n(サブネット数)	3
N_m(端末数)	5
ユーザ要求タイムアウト	5(秒)
コアデポ処理タイムアウト	100(ミリ秒)
複製デポ処理タイムアウト	50(ミリ秒)
シミュレーション時間	15分
P_w(更新が同じ端末で連続する確率)	90%
サブネット間パケット損失率	0-5%
サブネット内パケット損失率	0or1%
T_i(更新サブネットでの要求間隔)	5(秒)
T_s(上記以外での要求間隔)	5-60(秒)

表2 無線リンクでパケットロスがない場合の評価結果
Table 2 Simulation result(no packet loss at wireless link).

複製管理モデル→	移動なし	移動あり
平均更新時間 (ms)	28.4	24.1
更新成功数	170	165
更新成功確率 (%)	99.8	99.7
平均読出時間 (ms)	0.1	0.1
読み出し成功数	679	691
読み出し成功確率 (%)	100	100

移動なし: 最初にある端末にコアデポが作成されると，シミュレーションが終るまでその場所を固定とする．

移動あり: 更新要求の発生したノードにコアデポを移動する．ただし，移動はコアデポでの更新処理完了後，結果をユーザや要求元に返した後で行うこととする．また，コア移動は複製側から要求することとし，もしコア移動要求時に次の更新処理が開始されていたら，コア移動は行わないこととする．

4.1.4 その他のパラメタ

その他，シミュレーションで用いたパラメタを表1に示す．

4.2 評価結果と考察

最初に理想的な場合として，サブネット内，サブネット間とも無線リンクでのパケット損失が発生しない場合について評価を行った．表2に移動あり / なしモデルにおける更新及び読み出しの平均応答時間，アクセス成功数，及びアクセス成功確率を示す．各成功数はスループットの目安となる．ただし，T_s を60秒とした．

表2より，更新に関してはコアデポを移動した方が応答時間が短くなっていることが分かる．つまり，同じ端末で連続して更新が発生する場合，コアデポへ要求を転送する必要がないため応答時間は短くて済む．ただし，パケット損失が発生しないため，更新の成功数や成功確率には差が生じない．

一方読み出しの性能に関してはどちらのモデルも差がないことが分かる．ここで読み出し時間が極端に短いのは，基本的に無線リンクでパケット損失が発生しないため，読み出しは全て

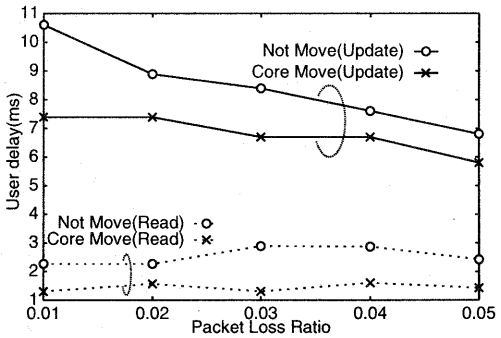

図4　パケット損失率と遅延の関係

Fig. 4　Access delay and packet loss ratio.

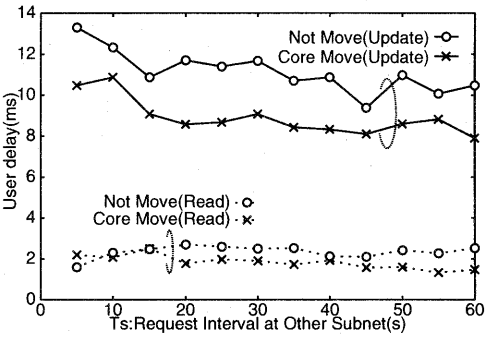

図7　要求間隔と遅延の関係

Fig. 7　Access delay and request interval.

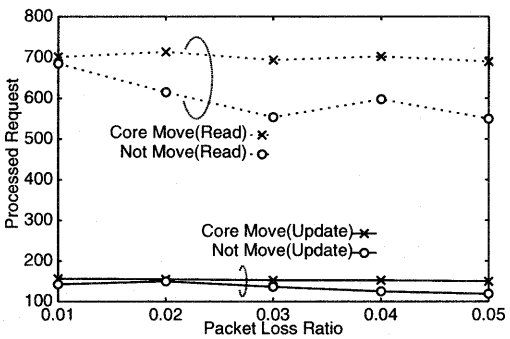

図5　パケット損失率とアクセス成功数の関係

Fig. 5　Number of success and packet loss ratio.

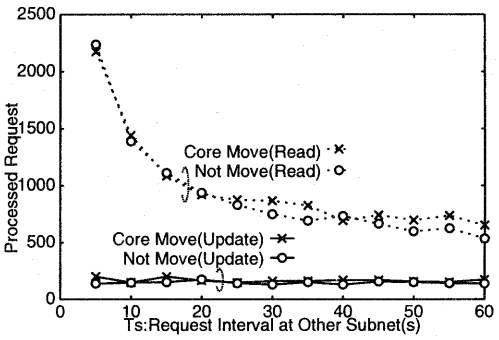

図8　要求間隔とアクセス成功数の関係

Fig. 8　Number of success and request interval.

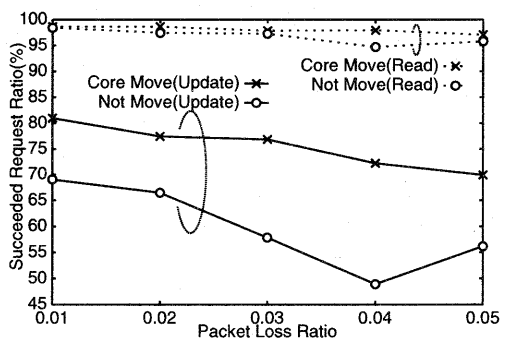

図6　パケット損失率とアクセス成功確率の関係

Fig. 6　Success ratio and packet loss ratio.

ローカルな複製データの読みだしで済んでしまうためである．

次に，無線リンクでパケット損失が発生する場合の評価を行った．サブネット内は比較的安定しているため1%のパケット損失が発生するものとする．この時サブネット間のパケット損失確率を1%から5%まで変化させた時の，移動あり / なしモデルにおける平均応答時間，アクセス成功数，及びアクセス成功確率を計測した．その結果を図4-図6に示す．なお，T_sを60秒とした．

図4より，どちらの方式もパケット損失が発生しない場合(表2)に比べて更新遅延が小さくなっていることが分かる．これは更新時に同期をとるためのパケットが損失して複製デポが無効となるため，次回以降の更新時に同期をとる複製デポの数が少なくなるためである．一方パケット損失が発生しない場合は，常に全ての端末上の複製デポと同期をとる必要があるため，更新遅延が大きくなっていた．

また，図4より，コアデポを移動した方が更新，読み出しとも遅延が小さくなることが分かる．これは，コアデポを移動した場合，データアクセス時にローカルな端末や同一サブネット内にコアデポが存在する確率が高く，端末間やサブネット間の通信が削減される効果と考えられる．

更に，図4～図6より，コアデポを移動した場合はパケット損失率が大きくなっても，コアデポを移動しない場合に比べてあまり性能に大きな影響を与えないことが分かる．これはコアデポが更新や読み出しの要求発生端末や同一サブネットに存在する確率が高いため，サブネット間の通信環境(パケット損失率)の変化の影響を受けないためと考えられる．

最後に，更新発生端末が存在しないサブネットでの要求発生間隔(T_s)を5秒～60秒と振らせた場合の，平均応答時間，アクセス成功数，及びアクセス成功確率を計測した結果を図7-図9に示す．

図7より，コアデポを移動した方が更新処理の遅延が小さくなることが分かる．これは，更新要求が発生した端末上にコア

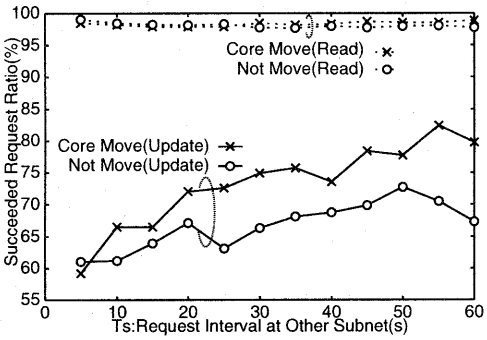

図9　要求間隔とアクセス成功確率の関係
Fig. 9　Success ratio and request interval.

デポが存在する確率が高いためである．また図9より，コアデポを移動した方が更新成功率も高くなっているのが分かる．特に，更新要求の発生していないサブネットでの要求間隔 (T_s) が大きくなると更新成功確率も高くなっているが，これは他のサブネットからの読み出し要求が少なくなるため，複製デポの数も少なくなり，同期をとる際に失敗する確率が小さくなるためである．

なお図8より，T_s が大きくなるとどちらの方式も読み出し成功数が低下しているが，これは要求間隔が大きくなると，要求発生数自体が少なくなるためである．

このように，無線リンクが不安定でサブネット単位での移動が発生し，連続的なデータの更新が発生する場合には，コアデポを移動させることによりデータアクセスの可用性を高めることが確認できた．

5. 関連研究

分散ファイルシステムでは，ファイルを弱整合なデータとして扱うものが多い．例えば2.2節でも触れたように，Coda [5] では，クライアント内にファイルのキャッシュを保持し，サーバと切断された際にはキャッシュに対する更新操作も許しているため，キャッシュデータの一貫性についてはネットワーク回復後まで保証されない．

分散DBにおいては強整合のデータも扱っているが，DBサーバとの切断時にサービスを継続するものは少ない [6]．一方 Wang と Pâris はデータ更新時に通信する必要があるノードの集合 (CSCR) を管理するレフェリを設け，あるノードをネットワークから切断する際に，CSCR としてそのノードのみをレフェリに登録することにより，切断時のデータ更新を許す方式を提案した [7]．しかしノードの切断があらかじめ予測できない場合，そのノードでのデータ更新は不可能である．

一方 CAOSS は更新時に全ての複製で同期をとるため，強整合のデータを扱うことが可能である．また，ネットワーク分断時にも複製データの読み出しを許すため，読み出しの可用性を高める．更にシミュレーション評価で確認したように，同一の端末で連続してデータの更新が発生し易い場合，予期せぬネットワーク分断 (シミュレーションではパケット損失率の増大) に起因するデータの可用性低下を防ぐことが可能である．

6. まとめ

将来のユビキタスサービスで必要な適応型オンラインストーレジシステム CAOSS における複製データの管理方式を提案し，シミュレーション結果を用いて有効性を示した．特に，物理的に近い場所で連続してデータアクセスが発生し易い場合に，複製データの更新権限を持つ「コアデポ」を移動させることにより，データアクセスに適応して可用性を高め，アクセス速度も小さくすることを示した．

CAOSS では弱整合な複製も扱うことが可能なため，今後はその詳細設計及び定量評価を行っていく．またプロトタイプシステムを構築して更に有効性を示していく予定である．

文　献

[1] http://www.blink.com
[2] Wael R.Elwasif, James S.Plank, Micah Beck and Rich Wolski, "IBP-Mail: Controlled Delivery of Large Mail Files", NetStore 99, Oct. 1999.
[3] 井上 知洋，中村 元紀，久保田 稔，"適応型オンラインストーレジにおける切断時データ同期方式"，FIT2002 M-88，2002年9月．
[4] 中村 元紀，井上 知洋，久保田 稔，"アドホックネットワーク環境のためのデータ管理方式"，信学総大 B-15-6，2002年3月．
[5] James J. Kistler and M. Satyanarayanan, "Disconnected Operation in the Coda File System", ACM Trans. on Comp. pp3–25, Feb. 1992.
[6] Daniel Barbará, "Mobile Computing and Database - A Survey", IEEE Trans. on Knowledge and Data Engineering, pp. 108–117, Jan. 1999.
[7] Q.R. Wang and J.F. Pâris, "Managing Replicated Data Using Referees", Proc. Workshop Mobility and Replication, ECOOP '95, Aug. 1995.