マルチメディア通信と分散処理 85－36
（1997. 11. 7）

スナップショットシステムの提案と試作
— WWW 情報空間への時間軸の導入 —

菅山 亨†, 知念 賢一†, 山口 英†, 尾家 祐二‡

†奈良先端科学技術大学院大学
‡九州工業大学 / 奈良先端科学技術大学院大学

概要

Internet を代表するサービスである WWW (World Wide Web) は情報提供者が増加したことにより新しい情報基盤となりつつある. WWW を用いた情報提供では, 一般にその利用者が知らないうちに内容の更新や削除が行なわれる。そのため利用者が過去に参照した情報を再び必要としても, もはやその情報にはアクセスできないことがしばしば起こっている. 本研究は, WWW で提供される情報を時系列にしたがって蓄積し, 蓄積された情報から利用者が必要とした任意のページ (スナップショットと呼ぶ) を取り出すことができるような機構を提供することを目的とする.

本稿ではまずスナップショットの概念を定義する. つぎに過去の情報を取り出すためのスナップショットシステムを提案する. さらに提案に基づきシステムの設計及び試作を行ない, 試用を通じて有効性について考察を行う.

The Snapshot System for World Wide Web

Tohru Sugayama†, Ken-ichi Chinen†, Suguru Yamaguchi†, Yuji Oie‡

†Nara Institute of Science and Technology
‡Kyushu Institute of Technology / Nara Institute of Science and Technology

Abstract

The WWW is one of major services on the Internet. The WWW contents are updated independently by the information provider, and the user doesn't know the contents are modified or deleted until he/she has accessed them. Therefore the users are no longer able to access the lost contents which they need. In our research, we design how to preserve the contents automatically in the sequence of time and build the mechanism to take them out later. We call the requested contents at a moment as a *snapshot*.
In this paper, we define the snapshot of the WWW and propose the snapshot system in order to take the preserved contents for the user. We also implement the prototype, and consider the system usability through the trial.

1 はじめに

Internetが爆発的に普及するにしたがってその上でさまざまな情報サービスが提供されるようになった. 特にWWW (World Wide Web) の普及は目覚ましく, Internetの代表的なサービスになっている. これにともない, 従来は新聞やテレビなどのメディアで別々に扱われていた情報をWWWを通じて提供する機会が多くなってきた.

WWWは情報の提供やその更新が容易に行なえる. ところがこのような性質により, 利用者が過去に取得した情報を再度取得しようとした際には必要とする情報が消失している可能性がある. こうした問題を解決するサービスとして情報のバックナンバを提供する情報提供者もいる. しかしすべての情報がバックナンバとして提供されているわけではない.

このような背景から, WWWにおいて利用者が過去に取得した情報を再び容易に取り出すことのできるシステムが要求されている.

本研究では, まず利用者が取得することのできる情報の単位としてスナップショットと呼ぶ概念を定義する. 次に前述した問題を解決するためにスナップショットシステムを提案する. さらにこの提案に基づいたシステムの設計及び試作を行ない, 実際に試用することによってその有効性を示す.

2 スナップショットシステム

WWWは情報更新が容易である反面, 利用者が以前取得した情報を再び閲覧することができない状況が問題となっている. 文献[1]によればオブジェクトの平均生存時間は約44日間であり, 約28%のオブジェクトが10日以内に更新されていると報告されている.

スナップショットシステムは以上のような問題を解決するシステムである. 利用者が閲覧した情報を蓄積し, 蓄積された情報から利用者が必要とする任意のページを取り出すことを目的とする.

2.1 スナップショット

本節では過去の情報を扱うための枠組みとしてスナップショットの概念について述べる.

WWWの情報の提供は各ページを単位に行なわれ, ページは文章や図などの複数のオブジェクトで構成されている. 本研究ではWWWで提供されるオブジェクトを時間軸を導入した空間で考察する (図1). この空間内で, それぞれのオブジェクトは独立して更新される. この空間を任意の時刻Tでのオブジェクトの集合をスナップショットと呼ぶ. このスナップショットの概念はページだけでなくWWW上のすべてのオブジェクトにあてはめることができる.

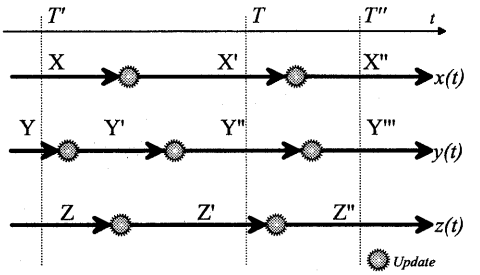

図 1: スナップショット概念図

2.2 オブジェクトの更新

利用者が図1に示されるページを時刻T, T', T''で閲覧を行なったものとする. この場合, 実際には存在するオブジェクトY'の存在を利用者は知ることができない.

スナップショットシステムを構築する際にはスナップショットをどこまで再現するのかが問題となる. つまりスナップショットを 1) 完全に再現する方法と 2) 不完全ながらも利用者にとって必要な再現性を確保する方法が考えられる.

1) の完全なスナップショットを再現するためにはWWW上の全てのオブジェクトを取得する方法が必要となる. この方法にはオブジェクトを定期的に監視し, その更新に合わせオブジェクトを取得する必要がある. 正確なオブジェクトの集合を取得するためには, 監視間隔をなるべく小さくする必要があるが実現は困難である. したがって, スナップショットを完全に再現するのは困難である. そしてこの考え方は本研究の目的と異なるため本研究では採用しない.

2) の必要な再現性を確保する方法では全ての過去のオブジェクトを蓄積する必要がなくなる. 本システムの目的は利用者が閲覧した過去の情報が重要となる. これは利用者にとっては閲覧したものだけが存在したものであり, 一度も閲覧していないものは存

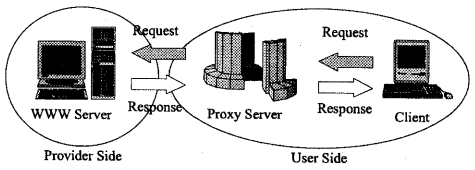

図 2: WWW 環境

在しなかったものとして考えられるからである. したがって, すべてのオブジェクトの更新を知る必要はない.

3 スナップショットシステムの設計

本章ではスナップショットシステムの設計を行なう. システムの設計にあたり以下の条件を設定し, その条件に従い設計する.

- 既存環境の変更を最低限に抑える
- HTTP(HyperText Transfer Protocol)[2] を用いて提供されている情報を対象とする

3.1 既存環境における位置づけ

代表的な WWW 環境は図 2に示すように WWW サーバ, 代理サーバそしてクライアントからなるモデルで表現できる. 本節ではこのモデルのどの部分にスナップショットシステムを構築すればよいかを議論する.

WWW サーバにおけるスナップショットシステム構築では, 過去に提供したスナップショットを構成するオブジェクトの蓄積を情報提供者が行えばよい. また WWW サーバにはスナップショットシステムの対象となるオブジェクトが保持されているため, スナップショットシステムがオブジェクトの正確な更新時間を把握することができる. このため, オブジェクトの更新を見逃すことがなく, 過去のスナップショットを正確に再現することができる. このような WWW サーバでのスナップショットシステムを実現するためにはすべての情報提供者がシステムを導入しなければならない. しかし情報提供者の方針によっては, このようなシステムが導入されない可能性がある. また WWW による情報提供自体が停止されると, その情報提供者が提供していた情報はすべて失われてしまう. 以上のように WWW サーバでのシステムの実現は情報提供者の意思や意図に左右され, 利用者にとって必要なスナップショットを構成するオブジェクトが蓄積されているとは限らない.

クライアントで構築する場合はクライアントに新たな機能を追加する必要がある. 現在広く用いられているクライアントは, 規模が大きく実行形式で配布されているため新たな機能を追加するのが困難である.

代理サーバで構築する場合は現在用いられている代理サーバを置き換えるか, あるいはシステムを追加するだけでシステム構築が可能となる. また同じ代理サーバを複数人で利用することによって蓄積情報の共有も可能となる.

以上の考察により本研究では既存の環境の変更が少ない代理サーバにおけるスナップショットシステムを構築する.

3.2 必要となる機構

スナップショットシステムを実現するためには以下の機構が必要となる.

- 情報を時間に沿って蓄積する機構
- 蓄積した情報から必要な情報を取得する機構

以下ではこれらの機構について記述する.

3.2.1 蓄積方法

利用者が閲覧したスナップショットに含まれるオブジェクトを時系列にしたがって蓄積するためには, そのオブジェクトを一意に示す識別子及び最終更新時刻を同時に保存する必要がある. WWW ではオブジェクトの位置を一意に示す URL (Uniform Resource Locators)[3] を使用しているため, オブジェクトの識別子として URL を利用する. オブジェクトの更新時刻も同時に保存しなければならないので, オブジェクトを取得する際に入手できるその最終更新時刻を使用する.

3.2.2 スナップショットの要求法

利用者がスナップショットを取得するためには本システムに対してクライアントから要求を送らなければならない. 本節では本システムとクライアントとの通信の実現法について述べる.

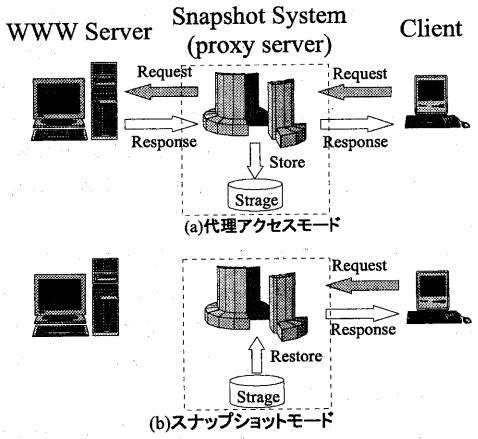

図 3: 代理サーバを用いたスナップショットシステム

システム動作モード

代理サーバを用いたスナップショットシステムは2つの動作モードがある (図 3). 一つは代理サーバモードでもう一つはスナップショットモードである. これらのモードはクライアントから本システムに要求を送ることで切り替える.

代理サーバモードでは本来の代理サーバとしての機能を提供する. スナップショットシステムはクライアントからの要求にしたがって WWW サーバからオブジェクトを取得する. 取得したオブジェクトをクライアントに返答し, 同時にそのオブジェクトを記憶装置に蓄積する.

スナップショットモードではクライアントからの要求にしたがって記憶装置から対応するスナップショットを構成するオブジェクト群を検索する. 必要なオブジェクト群が見つかった場合は要求されたスナップショットを構成するためにそれらのオブジェクト群をクライアントに送信する. また必要なオブジェクト群が見つからなかった場合はその旨クライアントに返答する.

実現法の比較

クライアントが本システムと通信を行なうためには新たにプロトコルを定義する方法と既存の方法 (URL や HTTP) を拡張する方法が考えられる.

新しいプロトコルを定義する方法はクライアントを改造する方法と, 現在のクライアントに変更を加えず新たなプロトコルによって制御するプログラムを作成する方法がある. クライアントの改造は困難

```
<scheme>:<sceme-specific-part>
(a) General form

<user>:<password>@<host>:<port>/<url-path>
(b) ip-scheme part

http://<host>:<port>/<path>?<searchpart>
(c) http url

http://www.aist-nara.ac.jp/index.html
(d) url with scheme

/images/japan.jpg
(e) absolute path

../../images/japan.jpg
(f) relative path
```

図 4: URL の表記法

であるので採用できない. また新しい制御用プログラムを作成する方法では既存の環境への影響はないが, 各プラットホーム毎に制御用のクライアントを作成しなければならないので採用しなかった.

次に既存の方法を用いる場合は, HTTP を拡張する方法と URL を拡張する方法が考えられる. HTTP を拡張する方法ではクライアントの機能拡張が必要となる. 一方 URL を拡張する方法では既存の環境を変更することなく実現できる.

以上の考察から本システムでは既存の通信方法に変更を加えず, 新しいクライアントの作成が必要とならない URL の拡張によってスナップショットを指定する. URL の拡張については後述する.

URL の拡張

URL の一般的な表記は図 4(a) となる. スキームには HTTP や FTP などがある. また IP を用いている場合はスキーム特定部分が図 4(b) のようになる. HTTP の表記を図 4(c) に示す. URL の拡張では新しいスキームの定義を含めユーザ, パスワード, ホスト, ポート, パス, サーチパートの 7 つの拡張方法がある. 新たなスキームの定義にはクライアントの改造が必要となる. 既存環境の変更を最小限に抑えるという前提からこの方法は好ましくない. したがって変更可能な箇所はユーザ, パスワード, ホスト, ポート, パス, サーチパートとなる.

HTML(HyperText Markup Language) 中での他のオブジェクトへのリンクはスキームを含めた URL 全体の指定 (図 4(d)), 絶対パス (図 4(e)) 及び相対パス (図 4(f)) の 3 種類で表記される.

```
http://#command=argumen#@host:port/path
```
(a)URLの拡張

```
hostport = ["#"command["="argument"#"]]host[:port]
command  = "date" | "current" | "first" | "last"
         |" index" | "search"
argument = *char | *digit["-" *digit]
```
(b)URLの拡張(BNF表記)

command	argument	remarks
date	date[-date]	specify date
first		oldest data
last		latest data
current		current data
index		make index
search	key[-key]	search

(c)拡張URLの命令と引数

```
http://#date=19961102#www.aist-nara.ac.jp/index.html
http://#forvar=foo#www.aist-nara.ac.jp/index.html
http://#index#www.aist-nara.ac.jp/index.html
```
(d) 拡張URLの例

図 5: URL の拡張

クライアントは相対パスや絶対パスを利用してWWW サーバに要求する URL を生成するので, パスに影響を与える拡張は困難である. すなわち,URL のパスやサーチパートの拡張は困難であるため本研究では採用しない.

したがって, 拡張可能な部分はユーザ, パスワード, ホスト, ポートとなる. そこでホスト部分に命令と引数を与えることで, スナップショットに含まれるオブジェクトを指定する. 図 5(a) に RFC1768 による URL の一般形を示し, 図 5(b) に今回の拡張を示す. また拡張した命令と引数を図 5(c) に示す. この拡張によって図 5(d) で示されるような URL による過去のオブジェクトが指定できる.

次に HTML に現れる URL 全体の表記の対処について述べる. URL 全体の表記には, スナップショットシステムで状態を持つことで対応する. 問題は時刻を指定していない要求オブジェクトの時刻をいかに設定するかであるから, クライアントからの要求を保持しておき, 時刻の指定がない場合には直前の要求の時刻を要求の時刻と見なす.

これらの処置によって, URL を意味拡張してクライアントの生成する全ての要求に時刻を付加し, スナップショットを指定することが可能となる.

4 実装

本方式の有効性を確認するために, 本システムのプロトタイプを実装した. プロトタイプの実装では

表 1: 評価環境

	機器	OS
スナップショットシステム	SS1000	Solaris 2.5.1
クライアント	PC	WindowsNT
WWW サーバ	PC	BSD/OS 2.1

Perl 言語を用いている. また実装環境としては Sparc Server 1000 上の Solaris2.5.1 を使用した.

現状では, 代理サーバとして動作しながらオブジェクトを時系列に沿って蓄積する部分と蓄積されたオブジェクトからスナップショットを取得しクライアントに返答する部分のプロトタイプの実装が終了している.

オブジェクトの蓄積には, 時系列に沿ってオブジェクトを保存する必要がある. この実現のため HTTP で規定する Last-Modified フィールドを利用している. ただし Last-Modified フィールドを付加しない WWW サーバも存在する. Last-Modified フィールドが利用できない場合, 代理アクセスによってオブジェクトを取得した時刻と共に保存している.

5 評価

実装したプロトタイプについて評価を行った.

表 1に示すような 3 つのホストを用意して, スナップショットシステム, クライアント, WWW サーバを動作させた. クライアントの代理サーバとしては, スナップショットシステムを指定した.

次に WWW サーバで提供されているページをクライアントで閲覧した. その際, 閲覧したページの内容が日付とともに蓄積されていることを確認した. WWW サーバ上でここで先ほど閲覧したページの内容の書き換えを行った. この状況でクライアントを用いて更新されたページの閲覧を行った. このときページの更新が反映されていることを確認した. ここで WWW サーバで更新直前の時刻の指定を行い更新以前のページが表示されることを確認した. 同様に更新と閲覧を数回繰り返し, 過去に閲覧した時点での情報が取得できることを確認した.

以上によりスナップショットシステムが正常に動作していることを確認した.

6 今後の課題

6.1 大規模化

代理サーバはWWW環境の中で負荷が集中する場所であり,多数の要求に対する耐久性が要求される.今回試作したシステムは多数の要求を処理することを考慮していない.今後,多数の要求を処理できるシステムの設計・実装を行なうことにより,本格運用に耐えうるシステムを構築する必要がある.

6.2 蓄積情報の破棄

本稿で述べたスナップショットシステムでは,オブジェクトの蓄積を行うだけで,その破棄については考慮していない.しかし運用を行なううえで利用できる資源は限られている.したがってオブジェクトの破棄を行なうことや,さらに大規模な記憶装置を導入することが求められる.

6.3 動的なオブジェクト

本システムではCGI(Common Gateway Interface)によって生成されたオブジェクトのように,アクセスの度に返される内容が異なるオブジェクトに関しては考慮していない.このようなオブジェクトによって構成されているスナップショットを正しく表示するにはURLのサーチパスも保存する必要がある.

またHTTPで規定されているClient Request HeaderのUser-Agentを判断してオブジェクトの内容を変化させたり,Cookieを用いて利用者に合わせた内容を選択し利用者に返答するWWWサーバが登場してきた.このようなWWWサーバに対応するためにはオブジェクトと共にClient Request Headerを保存する必要がある.

7 まとめ

WWW上で過去に閲覧した情報が失われても利用者の要求に応じてそれを閲覧可能とするために,スナップショットと呼ぶ概念を導入し,過去に閲覧した情報をスナップショットとして提供するシステムを提案した.閲覧情報の蓄積はWWWの代理サーバで行ない,スナップショットの指定のためにURLの拡張を行なった.スナップショットシステムを試作し,設計した機構が正しく動作することを確認した.

今後は大規模な要求に耐えうるシステムを構築するとともに機能の充実を図る.また長期間運用を行なうことでその問題点を明らかにし,改善を行なって行く予定である.

謝辞

本研究を行うにあたり、多大なご協力を頂いた奈良先端科学技術大学院大学情報ネットワーク講座の諸氏に感謝します。

参考文献

[1] A. Chankhunthod, P.B. Dangiz, C. Neerdaels, M.F. Schwartz and K.J. Worrell: A Hierarchical Internet Object Cache, Technical Report 95-611, Computer Sceince Department, University of Southern California, Los Angels, California (1995). (ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/HarvestCache.ps.Z).

[2] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee and others: RFC 2068: Hypertext Transfer Protocol — HTTP/1.1 (1997). ftp://ftp.internic.net/rfc/rfc2068.txt.

[3] T. Berners-Lee, L. Masinter and M. McCahill: RFC 1738: Uniform Resource Locators (URL) (1994). ftp://ftp.internic.net/rfc/rfc1738.txt.