ヒューマンインタフェース 71－6
（１９９７．３．６）

タイムマシン
- 時間情報に基づくWWWサイトの検索I/F -

小池雄一　杉浦淳　古関義幸

NEC C&C 研究所

現在、WWW 上に存在する大量の情報を整理し、ユーザが必要な情報を見つけ出すことを支援する手法の開発が急務となっている。情報を整理する様々な属性の中でも、時間は特に普遍的なものである。また、新聞・雑誌のバックナンバーサービスや音楽ヒットチャート等、時間情報で整理した情報を提供する WWW サイトも数多く存在する。これらの理由から、筆者等は WWW ページを整理する属性として時間情報が有効であると考え、時間順に並べられた大量の WWW ページを効率よく検索・表示する手法を開発した。筆者等の手法では、タイムスライダと呼ばれる UI を用いることで、ユーザは大量のページから必要なページを素早く選択できる。また、サーバ・クライアント間通信に用いられるプロトコルを設計した。このプロトコルはカスタマイズ可能であり、サービス提供者は、構築やメンテナンスのコストが低いものから、DB と連携した高度なものまで、様々なサーバー構成を取ることが可能である。

An Information Retrieval Method for Chronologically Ordered WWW Pages

Yuichi Koike　Atsushi Sugiura　Yoshiyuki Koseki

C&C Research Laboratories, NEC Corporation

There are several kinds of WWW sites, which arrange pages in chronological order, such as archives of online newspaper, magazine, and history maps. We call such services, **time search service**. Since time is one of the most important attributes used to find information, arranging pages in chronological order helps users find information. In this paper, we propose a method to construct time search services. For the client end, we use a user interface component, called **TimeSlider**. It is a kind of slider implemented as an ActiveX Control. TimeSlider helps users easily find information from many pages, since it displays a list of many pages in a small area. For the server end, we define a time-search-service protocol based on HTTP. The protocol is configurable and allows various kinds of server implementations. We constructed an example system, a CD sales chart archive, based on the method.

1 はじめに

WWW 上の情報サービスを利用する上で、ユーザが大量の情報の中から必要とする情報を選択しなければならない場面は非常に多い。大量の候補からの選択はユーザにとって困難な作業であるため、カテゴライズやフィルタリングといった、ユーザが一度に接する情報の数を削減する技術が開発されてきた[1]。しかし、これらの技術では対応が困難な状況も存在する。例えば、あるライブラリのオンラインリファレンスには数千の関数があり、一つのカテゴリにも数百のエントリが存在する。また、AltaVista [2]や Lycos [3]といったキーワード検索サービスでは、いくつかのキーワードで絞った後でもなお数百の候補が存在することがある。従って、候補を削減するだけでなく、ユーザが大量の候補の中から必要な情報を選択するのを支援する技術の開発が重要である。

大量の候補を表示する際には、候補を何らかの属性にしたがってソートして表示することが有効である。ソートに用いる属性としては、名前、時間、ファイルの種類、ファイルサイズ等が考えられるが、本稿では特に時間情報に注目する。時間に注目する第一の理由として、時間情報が人間が情報を検索する際の手がかりとして比較的頻繁に用いられることが挙げられる。第二の理由としては、時間という属性の普遍性が挙げられる。WWW 上には、オンライン新聞、雑誌、音楽ヒットチャート、ニュースリリースの様に、定期的に情報が更新されるサイトが数多く存在する。これらのページは必ず時間属性を持つため、インタネット上には時間で整理された情報を持つ多数のデータベースが潜在しているといえる。実際、いくつかのサイトでは、バックナンバサービスという形で時間に基づいた情報提供をおこなっている。また、バックナンバサービス以外にも、例えば歴史地図のように時間情報で整理することが有効な情報の提供形態も幾つか存在する。本稿では、このように WWW サイト上の情報を時間情報に基づいて整理し提供するサービスを総称して時間軸検索サービスと呼ぶ。

ここで、現存する時間軸検索サービスの一般的な構成を図1に示す。左側は、存在するページの時刻と概要のリストである。リスト上で項目を選択すると、その時刻に対応するページを見ることができる。また、各ページには「前へ」「次へ」リンクがあり、それを用いて連続するページを次々にブラウズすることが可能である。

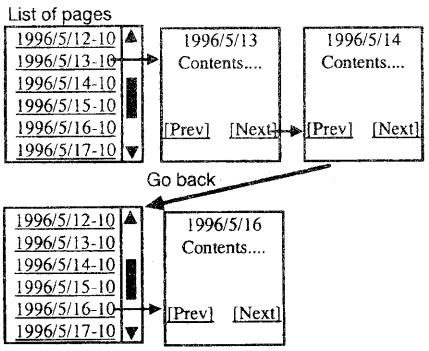

図1 バックナンバサービスの一般的な構成

この方式の問題点の一つにブラウジングがある。ユーザは、見るページを切り替えるたびにリスト画面に戻る必要がある。連続するページのブラウジングには、「前へ」「次へ」リンクが用意されているが、それ以外のページを見るには、ユーザがリストページと内容ページを交互に移動しなければならない。これの解決策として、Frame 機能を用いてリスト画面と内容のページを同時に表示する方法があるが、二つのページを同時に表示する場合にはリスト画面の面積は限られる。このため、ページ数が多くなると、一度に表示できるリスト項目数が少なくなり、ユーザの使い勝手は低下する。また、全てのページに「前へ」「次へ」リンクを設けるため、余分な作業が発生する、ページレイアウトに影響を及ぼす、等の理由でサービス提供者から見ても好ましいとは言えない。

本稿では、これらの問題を解決する時間軸検索サービスの実現方式を提案する。筆者等は、クライアントのユーザビリティ向上のため、**タイムスライダ**という UI コンポーネントを開発したタイムスライダは、小さい面積に大量のページのリストを表示することが可能、キーワードフィルタリングと統合された UI を持つ、などの特長を持つ ActiveX コントロール[4]であり、ユーザは大量の候補の中から必要な情報を効率よく選択することが出来る。また、筆者等は時間軸情報サービスにおけるクライアント・サーバ間の通信プロトコルを設定した。このプロトコルはカスタマイズ可能であるため、サービス提供者は時間に対応したディレクトリにファイルを置いただけの簡易なサーバから、CGI と DB を連携させた高度なサーバまで、様々な実装を選択できるようになっている。

次章以下では本研究で提案する時間軸検索サービスのアーキテクチャについて述べる。

2 アーキテクチャの概要

時間軸検索サービスといっても、様々な形態のものが考えられるが、本研究では、定期的に更新されるページの過去や未来の状態をそのまま再現する「タイムマシン」型を対象とする。これに対し、例えば時間と共に変化する情報を、一枚のグラフなどの形で見せる「非タイムマシン」型のサービスは対象としない。図2に本研究の手法に基づいて構築された時間軸検索サービスの概要を示す。

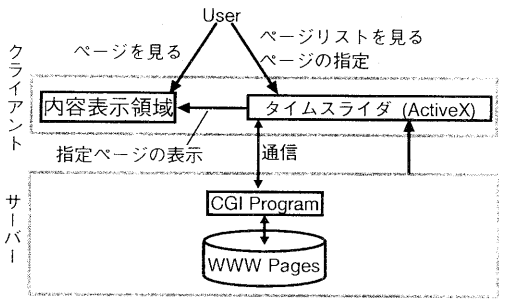

図2 時間軸検索サービスの概要

クライアントは、タイムスライダと呼ばれるユーザインタフェースコンポーネントと、ユーザが要求したページを表示する領域の二つから構成される。Frame を用いることにより、タイムスライダとページ表示領域は同時に表示される。タイムスライダは、ユーザが時間軸検索サービスにアクセスした時に自動的にサーバからダウンロードされる。ダウンロードされると、タイムスライダは後述するプロトコルに従ってサーバと通信を行い、サーバに蓄えられたページの時刻や概要のリストを取得しユーザに提示する。ユーザがタイムスライダ上で時刻を指定すると、タイムスライダはその時刻に対応するページをプロトコルに従ってサーバに要求し、その結果を JavaScript モジュール[5]を介してページ表示領域に表示する。

サーバは、クライアントのタイムスライダからプロトコルに従って送られてくる要求に従い、ページのリストや概要情報、要求されたページ等をクライアントに返却する。

3 クライアントの構成

時間軸検索サービスとは基本的に、ユーザが時刻を指定すると、その時刻に対応するページをユーザに提示するものである。ユーザがより効率良く時間軸検索サービスを利用するには、これに加え、以下に述べる機能が重要となる。

1. ブラウジングの支援
   リストのページに戻らずに複数ページを、見ることが出来なければならない。
2. 概要の表示
   ユーザが求めるページを特定するのに、時刻情報のみでは不十分な場合がある。このため、ページの概要を合わせて表示する等の機能が必要となる。
3. キーワードによる検索
   時刻、概要に加えて、キーワードによる検索が可能であることが望ましい。ただし、キーワード検索をしたためにユーザインタフェースが大きく変化することは望ましくない。キーワード検索機能が、通常のユーザインタフェースからシームレスに利用できる必要がある。

図3に、時間軸検索サービスのクライアント構成例を示す。WWW ブラウザはフレームによって二つに区切られ、左側にタイムスライダが、右側にタイムスライダでユーザが指定したページが表示される。

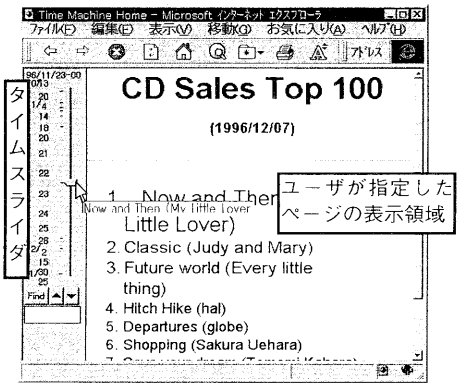

図3 クライアントの構成

3.1 タイムスライダ

タイムスライダは、ユーザインタラクション、サーバとの通信という二つの処理を行なう。UI コンポーネントとしてのタイムスライダは、大量のリストを少ないスペースに表示し、ユーザが必要な情報を効率よく選択するのを支援する。タイムスライダの左側には、早い時刻が上になるように時間を示す目盛りが表示される。目盛りの脇には小さいマークが打たれており、サーバ上にページが存在する時刻を示す。右側のつまみを動かすことで時刻を指定することが可能である(図5)。通信コンポーネントとしてのタイムスライダは、ユーザが指定した時刻から最も近くにあるページをプロトコルに従ってサーバ

に要求し、それを WWW ブラウザ上に表示する。

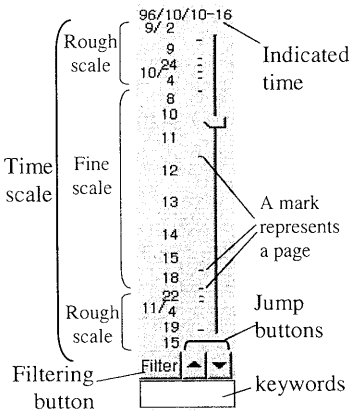

図 4タイムスライダ

3.2 時間目盛りを用いた時刻指定

タイムスライダの時間目盛りは、以下の三つの特長を持つ。

1. **小マークによるページ位置の表示**
   時間軸上のどこにページが存在するかをユーザに示すことで、ユーザは効率良く時間軸検索サービスを利用可能となる。従来の手法では、各ページ毎にそのページの時刻と概要が表示されていたが(図1)、タイムスライダでは、時間目盛り上でページが存在する箇所に小さいマークを打つことでこれを示す(図4)。このため、より少ない面積で多くのページを表示することができる。
2. **非線型の目盛り**
   一度に表示される時間範囲が広いほど、ユーザが必要とするページを見つけやすくなるが、細かい粒度での時刻指定が不可能となってしまう。これを解決するため、タイムスライダでは、中心部分では細かく、端の部分では粗いという、非線型の目盛りを利用している(図4)。例えば図5に示したタイムスライダでは、中心付近では 1 ドットが約 1 時間に対応するが、端に行くに従って粒度が粗くなり、最も端の部分では 1 ドットが約 2.5 日に対応する。タイムスライダ全体で約 4 ヶ月分の時間を表示できる。
3. **ユーザ操作に応じた目盛りの移動**
   あるページを指定するには、つまみを動かしてページを示すマークの所まで持って行けば良い。しかし、必要とするページが目盛りの粒度の粗い端の方にある場合には、これは困難な作業である。また、そもそも必要とするページが現在表示されていない可能性もある。このため、タイムスライダでは時間目盛りがつまみの位置に応じて移動する。ユーザがつまみを掴んでいる時、つまみが目盛りの中心より上にあるならば、すなわち中心よりも早い時刻を指しているならば、目盛りは下に移動する。逆につまみが中心より下にある場合、目盛りは上に移動する。この時目盛りの移動速度は、中心の時刻とつまみの指している時刻の差に比例する(図5-b)。この機能により、ユーザが指定したい時刻が目盛りの粗い部分にある場合、目盛り中心付近に移動させてから時刻を指定することができる。このタイムスライダの機能は、ユーザには自分の指定したい時刻付近につまみを持って行くと、そこの時刻を中心に持ってくるように目盛りが移動するように感じられる。

以上の特長により、タイムスライダを用いることで、ユーザは広い時間範囲の中から細かい時間粒度で効率よく任意の時間を指定することができる。

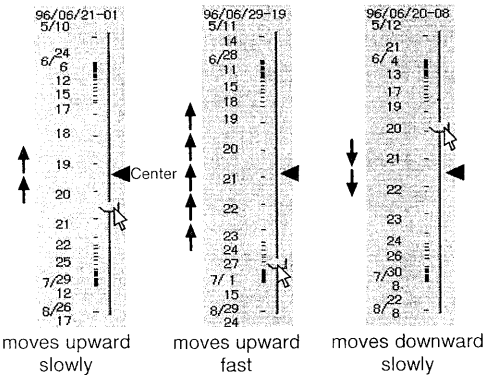

図5 時間目盛り

3.3 概要の表示

本研究の時間軸検索サービスでは、各ページの概要を時刻と共にユーザに提示し、目的のページを特定するのを支援する。全てのページの概要を一度に表示すると多くの面積を必要とするため、タイムスライダではユーザが注目するページの概要のみを動的に表示する。ページの概要は、サーバに蓄えられており、時刻のリストと同時に取得される。ユーザがつまみをつかむと、小さなウィンドウがポップアップし、その時点でつまみが指しているページの概要が表示される(図6)。その状態でつまみを動かすと、ウィンドウの内容が次々に変化するため、ユーザは複数のページの概要を次々にブラウズすること

ができる。

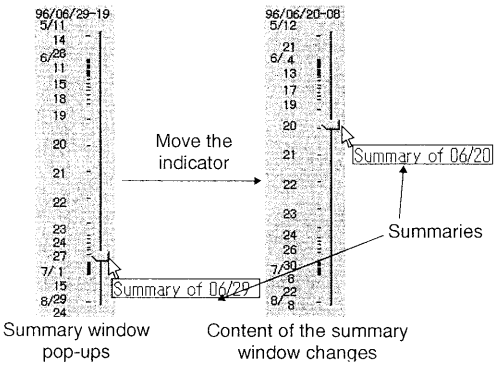

図6 概要の表示

3.4 キーワード・フィルタリング

図4に示した"Filter"ボタンの下にあるテキストフィールドにキーワードを入力してボタンを押すと、タイムスライダ上にはキーワードを含むページのマークだけが表示される。この機能により、キーワード・フィルタリングと、時間軸検索サービスとが単一のユーザインタフェースで統合されることになり、キーワード・フィルタリングを行うとインタフェースが変化してしまうようなシステムに比べ、ユーザビリティが高いと言える。

3.5 ジャンプ

多くの時間軸検索サービスでは、各ページには「前へ」「次へ」リンクがあり、リストページに戻ることなく連続するページをブラウズすることができる。本方式でも、図4に示すジャンプボタンを用いて連続するページをブラウズできる。キーワード・フィルタリング後にページの間隔が開いてしまい、つまみでは効率良くブラウジング出来ない場合に特に有効となる。

4 サーバの構成

サーバは、以下の 2 つの部分から構成される。サーバ・クライアント間プロトコルがカスタマイズ可能であるため、サーバの構成もそれに対応して様々な形態を取ることが可能となっている。

1. **ページアーカイブ部**
   ページアーカイブ部は、WWW ページを蓄積し、要求されたページをクライアントに返却する。筆者等の方式では、ページアーカイブ部の構成はカスタマイズ可能である。図7の左に示すようにページを格納した DB と CGI プログラムの組みでも良いし、図7の右に示すように時刻に対応するディレクトリに置かれた HTML ファイルでも良い。前者の場合、CGI プログラムはプロトコルに従って送られてくるクライアントからの要求に対応する必要がある。また、後者の場合はやはりプロトコルに従ったディレクトリ構造を持つ必要がある。
2. **ページリスト部**
   ページリスト部は、サーバ上に存在する WWW ページのリストをクライアントに通知する機能を持たなければならない。これに加え、ページの概要を通知する機能、キーワードフィルタリングした結果のリストを通知する機能などをオプションで持つことになる。

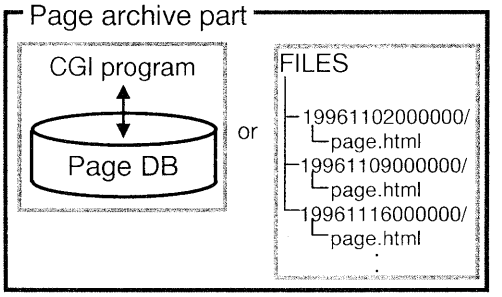

図7 ページアーカイブ部の構成

筆者等の方式の、時間軸検索サービス提供者の視点から見た利点の一つは、「前へ」「次へ」リンクの扱いである。従来の方式では、ユーザがページを次々に移れるようにするために、各ページに前後ページへのリンクが必要であった。このためサービス提供者には、各ページにリンクを付加する作業が発生する、各ページのレイアウトが前後リンクによって制約を受ける、等の問題点があった。また、キーワードフィルタリングを行なってページの前後関係が変化すると、各ページに固定的に付けられた前後リンクは無効になる。このため、前後リンクを CGI プログラム等で動的に生成する必要があった。これに対し本方式では、タイムスライダがページのブラウジング機能を持つため、各ページの前後リンクは不要である。

また別の利点として、サーバの構成がカスタマイズ可能であることが挙げられる。このため、サービス提供者は、構築が容易なサーバ構成から、高機能なサーバ構成まで、目的に応じたさまざまな形態の時間軸検索サービスを選択することが可能である。また、最も簡単なサーバ構成であっても、タイムスライダによる効率の良い検索 I/F を利用することが可能である。

5 時間軸検索サービスプロトコル

筆者等の時間軸検索サービスの方式では、クライアントとサーバは、HTTP 上に規定された時間軸検索サービスプロトコルに従って通信を行なう。このプロトコルは第一に、サーバ側の構成をカスタマイズ可能にすることを目的としている。また、サーバ・クライアント間の通信を最小限に抑えるため、

1. 時間軸検索サービス起動時に時刻・概要リストを取得
2. キーワード・フィルタリング時に時刻・概要リストを取得
3. ページ指定時にページ内容を取得

の状況でのみクライアント・サーバ間通信を行なう(図8)。このため、ユーザが通信によって中断されることなく、タイムスライダによるブラウジングが可能である。

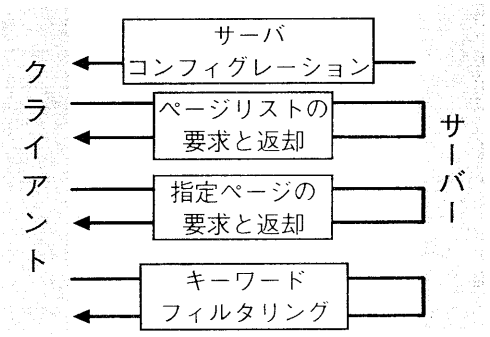

図8 サーバ、クライアント間の通信

5.1 サーバ・コンフィグレーション

サーバ・コンフィグレーションは、時間軸検索サービスを提供する各サーバの特性をクライアントに示すものである。サーバ・コンフィグレーションは属性名と属性値のペアのリストとして表現され、タイムスライダがサーバからクライアントにダウンロードされる時に、タイムスライダのプロパティとして渡される。以後、タイムスライダはサーバ・コンフィグレーションに従ってサーバとの通信を行うことになる。以下の表にコンフィグレーション対象の属性を示す。

1. **FilteringSupport:** キーワード・フィルタリング機能をサポートするかどうか
2. **SummarySupport:** ページの概要のリストを返却するかどうか
3. **FilteringURL:** ページリストをサーバから取得するURL
4. **RetrieveByCGI:** ページを取得するのにCGIプログラムを使用するか
5. **RetrieveURL:** 指定したページをサーバから取得するURL
6. **RetrieveFile:** ページがファイルとして置かれている時のファイル名

5.2 時刻・概要リストの取得

サーバが指定した特定の URL にアクセスすることで、クライアントはサーバにあるページの時刻・概要リストを取得することができる。また、サーバ・コンフィグレーションの FilteringSupport 属性が true の場合には、ユーザはキーワードによるページのフィルタリングを行うことができる。つまり、キーワードを引数とした URL にアクセスすることで、キーワードを含むページの時刻・概要リストを得ることができる。

サーバからクライアントに返却する時刻・概要リストの形式を図9に示す。一ページ分の時刻・概要リストは、'¥0'で終了する文字列形式である。文字列の先頭には時刻が配置され、その後に区切りを示す空白、最後に概要が置かれる。この形式の文字列がページの数だけ並べられる。クライアントでは、このデータから時刻・概要情報を取得する。

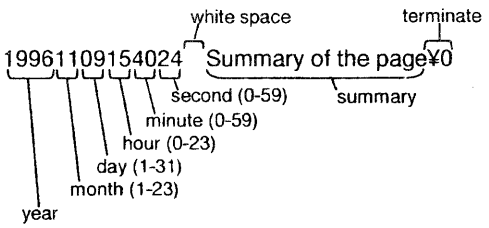

図9 時刻・概要リストのフォーマット

5.3 ページの取得

ある特定の時刻に対応したページを取得する URL は、時刻情報とサーバ・コンフィグレーションの属性を用いて作成する。図に、ある時刻に対応したページを取得する URL を生成する方法を示す。RetrieveByCGI 属性が true の場合、URL は時間情報を引数として CGI プログラムを呼び出す形式になる。RetrieveByCGI 属性が false の場合、URL は時刻情報を含んだディレクトリ下のファイルを取得する形式となる。

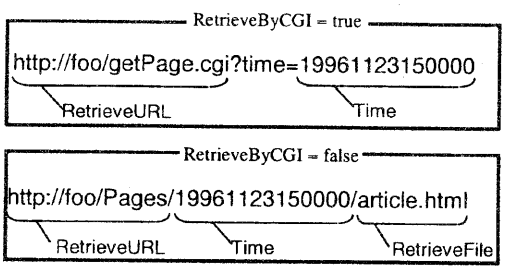

図10 ページ取得URL

6 インプリメンテーション

以上で述べた手法を用いて、CD ヒットチャートの時間軸検索サービスを実現するシステムを構築した(図3)。このシステムでは、毎週の CD の売り上げ Top100 のデータを 2 年分検索できる。サーバは以下のような機能を持つ。

1. 時刻・概要情報が表示可能
2. キーワード・フィルタリング可能
3. 指定されたページは CGI プログラムを用いずに返却

現在、このシステムのサーバ機能は、以下に示す二つのプラットフォームで実装された。

	実装 1	実装 2
HTTP サーバ	NCSA/1.5.2	IIS 2.0
OS	Unix (SVR4)	WinNT4.0
ページリスト[1]	CGI (シェルスクリプト 16 行)	ISAPI 準拠の DLL (C++140 行)
アーカイブ	ファイルに蓄積	ファイルに蓄積

7 関連する研究

WWW サイトの情報を整理・検索する手法としては、nif-T-nav [6]、WebGUIDE [7] が挙げられる。nif-T-nav は、WWW サイトのディレクトリ構造をツリー状にに表示することでユーザをナビゲートするシステムである。この手法では、クライアントでツリー構造を扱うため、通常の WWW アプリケーションに比べ、容易に階層構造の中から目的とする情報にたどり着くことが可能である。しかし、この手法は階層的な分類が困難な分野には適していない。この場合には大量の情報を一度に見せてそこから目的の情報を選択させる本手法が適すると思われる。

WebGUIDE は、過去の時点の WWW ページの内容や WWW サイトの構造と、現在のものとの差異をユーザに提示するシステムである。この手法は、時間と共に内容が変化するページを対象にユーザをナビゲートする点が本研究と共通している。しかし、WebGUIDE は例えば、What's New のコーナーのように、時間と共に少しずつ内容が変化するようなページをナビゲートするのに適しており、オンライン雑誌や新聞のように、時間と共に内容が大きく変化するものには適していない。これに対し、本手法では過去の情報をそのまま提示するため、オンライン雑誌などのバックナンバサービスに適すると言える。

大量の情報から求めるものを選択するユーザインタフェースとしては、AlphaSlider [8]、WING [9]が挙げられる。AlphaSlider はスライダを改良したものであるが、本方式のように目盛りが移動しないため、辞書のインデックスの様に目盛りが固定範囲のものには適しているが、時間のように目盛りの範囲が可変なものには適さない。

WING の Index View は情報を縦に並べたリストボックスの一種である。横方向にマウスを動かすことでリストの詳細度を変化させ、縦方向にマウスを動かすことでリスト上の項目を選択する。このため、やはり大量の情報からの選択に適している。Index View では、縦に並んだリスト項目を指定するために、横方向にもマウスを使う必要が出てくるため、操作が直感的ではない。このため、このインタフェースに熟練していないユーザにとっては使いこなすことが容易でない。これに対し、本方式ではマウスを縦方向のみに操作するため、インタフェースに熟練していないユーザでも効率良く用いることが出来る。WWW アプリケーションでは、アプリケーションにあまり接したことのないユーザが殆どであるため、インタフェースに熟練してないユーザでも効率よく使える本方式が適していると言える。

8 今後の課題

今後の課題の一つに、時刻・概要リストの取得方法の改良が挙げられる。タイムスライダは、時間軸検索サービスの起動時、及びユーザがキーワードフィルタリングを行った後に、サーバに時刻・概要リストを要求する。現在の実装では、全ての時刻・概要リストを取得するまで、ユーザはタイムスライダの操作を行うことが出来ない。このため、時刻・概要リストが大量になる時はユーザ操作が不可能な時間が長くなる。これを解決するため、時刻・概要リストを非同期に取得し、リスト情報が来た順にタイムスライダの表示に反映させ、ユーザの待ち時間

[1] キーワードフィルタリング機能を含む

を減らすことが考えられる。これを実現するには、タイムスライダが示している時刻に近いものからリストが送られてくるように、クライアント・サーバ間通信プロトコルの拡張する必要がある。

また別の課題として、時間以外の情報でWWWページを検索するシステムにタイムスライダの技術を応用することが考えられる。タイムスライダは、時間情報で整理された大量のページから一つのページを選択したり、複数のページをブラウズするのに適した技術である。このため、時間目盛りの代わりにアルファベットを使って大量のリストを表示することで、タイムスライダの技術をこれらの分野に応用することができると思われる。

9 まとめ

本稿では、時間軸検索サービスを構築する手法について述べた。時間軸検索サービスは、時間をキーにして整理された情報をWWW上で提供するサービスである。筆者等の手法は二つの特長を持つ。一つは、クライアントサイドをタイムスライダというユーザインタフェースコンポーネントで構成したことによるユーザビリティの向上である。タイムスライダは、時間軸を目盛りにとったスライダの一種である。中心付近では粒度が細かく、端の方では粗くなるように目盛りを表示することで、一度に広い時間範囲を表示させつつも、細かい単位での時間指定を可能である。また、タイムスライダの時間目盛りは、ユーザが指定しようとしている時刻を中心付近に持ってくるよう、ユーザ操作に応じて移動するため、ユーザはタイムスライダ上に示された任意のページに効率よくアクセスすることができる。これらのユーザ・インタフェースはタイムスライダをActiveXコントロールとして実装したことにより実現される。もう一つの特長は、サーバの構築作業の軽減である。本手法では、タイムスライダにブラウジング機能があるため、各ページに「前へ」「次へ」リンク等を設ける必要が無い。このため、各ページの内容に手を加える必要がない。また、タイムスライダはある一定のプロトコルでサーバにアクセスするが、最小のサーバ構成では、サーバでは時刻を名前として持つディレクトリに各ページを配置し、そのリストを作るだけで時間軸検索サービスを提供することができるようになる。

また、過去の音楽ヒットチャート情報を提供する時間軸検索サービスを本研究の手法に基づいて構築し、このシステムを用いて、時間情報に基づいた検索、キーワードによるページのフィルタリング、連続する情報の効率よいブラウズ、等の機能が働くことを確認した。

参考文献

[1] Ben Shneiderman, "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations," In *Proceedings of 1996 IEEE Symposium on Visual Languages*, pp. 336-343, 1996.

[2] AltaVista, URL: *http://www.altavista.digital.com/*

[3] Lycos, URL: *http://www.lycos.com/*

[4] Microsoft corporation, "ActiveX Technology and ActiveX Controls, ", 1996.

[5] Netscape corporation, "JavaScript Authoring Guide", 1996.

[6] Kirsten L. Jones, "nif-T-nav: A hierarchical navigator for WWW pages," In *Proceedings of the Fifth International World Wide Web Conference*, pp. 1345-1353, 1996.

[7] Fred Douglis, Thomas Ball, Yih-Farn Chan and Eleftherios Koutsofios, "WebGUIDE: Querying and navigating changes in Web repositories," In *Proceedings of the Fifth International World Wide Web Conference*, pp. 1335-1344, 1996.

[8] Christopher Ahlberg and Ben Shneiderman, "The Alphaslider: A Compact and Rapid Selector," In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)*, pp. 365-371, 1994.

[9] Toshiyuki Masui, Mitsuru Minakuchi and George Borden, "Multiple-View Approach for Smooth Information Retrieval," In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'95)*, pp. 199-206, 1995.