インターフェイスの街角 (26)– LensBar

増井 俊之

大量のデータを計算機の小さな画面で扱うためには、表示すべきデータを選択する手法（フィルタリング）と、データを効果的に視覚化して眺める手法（ブラウジング）が重要です。今回は、この 2 つの手法を効果的に統合した GUI ツール ``LensBar''を紹介します。

大きなデータを扱う手法

計算機の画面サイズは限られているので、膨大なデータを扱わなければならない場合であっても画面にはごく一部しか表示できません。したがって、なんらかの方法で表示量を制限したり、あるいは表示部分を移動させることによって、小さな画面で大きなデータを扱えるようにする必要があります。そのために、文字端末インターフェイスや近年のグラフィカル・インターフェイスではさまざまな工夫がおこなわれています。

計算機で大量のデータを扱う手法としては、従来から以下のような方法が利用されています。

データの一部を見る方法

文字端末では表示可能な文字数が限られていますから、テキストエディタなどでは文章の一部だけを画面に表示して編集するのが普通です。たとえば Emacs では画面を複数に分割し、複数のファイルを同時に眺めたり、同じファイルの異なる部分を眺めたりできるようになっています。表示するだけで編集しない場合には、more や less などのツールを用いて表示位置を変更しながら順次内容を眺めていきます。

グラフィカル・インターフェイスでは、表示位置をキー操作で変えるのではなく、マウスとスクロールバーを使って変更するのが一般的です。この方法には、全体のどの部分が表示されているかがノブの位置で直感的に分かるという利点があります。

いずれの場合も、ユーザーの操作によって表示位置を変化させて大きなデータを扱う点は共通しています。

データ構造を利用する方法

特殊なデータ構造を利用し、同時に扱うべきデータの量を制限する手法もよく使われています。最近は、UNIX をはじめとするさまざまな OS で膨大な数のファイルが扱われるようになりました。これについても、階層的なファイル構造を適用すれば各ディレクトリ内のファイルの数を制限できるので、ファイルを一覧しやすくなっています。

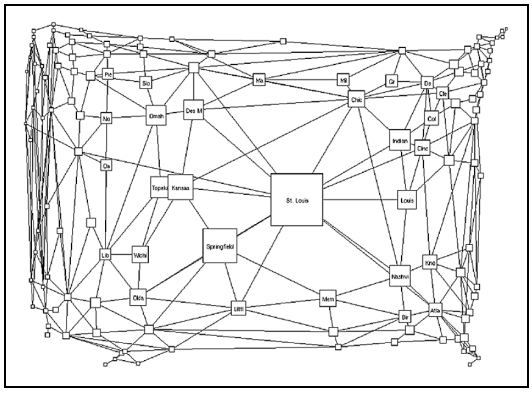
ただし、この方法では、全体のファイル構造と自分の位置をつねに把握しておかないと迷子になるおそれがあります。そのため、たとえば UNIX では pwd や ls などのコマンドがよく使われます。

表示すべき Web ページのデータが膨大でスクロールでは対応しきれない場合、Web ページを複数のページに分割して相互にリンクを張るといった手法もよくみかけます。これも、データ構造を応用した表示手法の一種と考えてよいでしょう。

フィルタリングによってデータを絞り込む方法

均質なデータが大量に格納されている電話帳やデータの相互関係が複雑に入り組んでいる Web のリンクなどについては、これらを扱うための適当なデータ構造がありません。そこで、キーワードなどの検索条件にもとづくフィルタリングを用いて表示するデータの量を制限することがよくおこなわれています。データベースや Web のサーチエンジンによる検索は、フィルタリング手法の一種といえます。

図 1　Focus+Context



全体と部分を同時に表示する方法

スクロールバーを利用している場合、現在表示されている部分が全体のどのあたりかを知ることはできますが、表示されていない部分に何があるかは分かりません。しかし、全体の構造を把握しつつ部分データを操作したいこともあります。そのために、表示されている情報だけでなく、全体も見えるようにする方法が考案されています。

1 つは〝Overview + Detail〟と呼ばれる手法で、全体構造をたとえば目次のように別ウィンドウに表示します。もう 1 つは〝Focus + Context〟で、場所によって表示の粒度を変える方式です。詳しく見たい部分は細かく表示し、概要だけ見えればよい部分は不要なものを間引いて表示します。

図 1 は、Focus + Context の手法で米国の地図を表現したものです [1]。ここでは地図の見たい部分（セントルイス付近）だけを拡大し、周囲を小さく歪ませて表示しています。

全体表示と部分表示とを簡単に切り替えられるのならば、これらを同時に表示するのと同様な効果が得られます。1998 年 7 月号で紹介したズーミングインターフェイス・システムでは、ユーザーがズーミング操作によって詳細表示と全体表示を滑らかに変化させ、データの全体構造と部分の詳細を同時に把握できます。

ブラウジングとフィルタリングの融合

大規模なデータの一部だけを操作したり眺めたりするには、必要な部分をうまく選択して表示・操作できるようにする技術が重要です。そのためには、以下のような機能が必要です。

図 2　Windows95 の TreeView と ComboBox

- 表示部分を滑らかに移動させる機能
- 表示したい部分を位置で指定する機能
- 表示したい部分を名前で検索する機能
- 重要な部分だけを選択的に表示する機能
- 可視部分とデータ全体との関係を明示する機能

検索システムや grep コマンドなど、データを検索したりフィルタリングしたりする手法は文字ベースのインターフェイスでもひろく使われています。一方、スクロールバーやズーミング・インターフェイスによって表示領域を滑らかに変化させる各種のブラウジング技術は、最近の GUI では一般的になっています。しかし、両者をうまく融合した GUI 手法には、まだこれといったものがありません。

Windows 9x で階層的ファイルシステムをブラウズするためには、ディレクトリごとにウィンドウを開いたり、TreeView というコントロールを用いたツールを使うのが一般的です（図 2 左）。大きなファイルシステムを眺める場合、ディレクトリごとにウィンドウを開いていくと、ウィンドウの数が多くなって扱いにくくなります。TreeView では、階層的データもある程度は扱いやすくなります。しかし、ディレクトリ内に多くのファイルがあるときなどは、全体を眺めるためにスクロールバーを多用しなければならず、階層を上下する操作もそれほどスムーズではありません。

■ TreeView や ComboBox はコントロールであり、ツールとはいえないのではないでしょうか?

リスト内の項目をメニューで選択する場合は、階層メニューや ComboBox コントロールが使われます（図 2 右）。数個ならともかく、数百個の項目のなかから目指すものを選びだすのは容易ではありません。

一般的なスクロールバーや階層メニュー、TreeView、ComboBox は、データの一覧のなかから必要なものを選ぶという目的はほぼ共通していますが、機能が低いため

に、場面に応じて異なるツールを使い分けなければなりません。

LensBar

LensBar[2, 3] は、さきほど挙げた必要な機能をすべて備え、フィルタリングとブラウジングをうまく融合させたGUI 部品です。比較的単純なツールであるにもかかわらず、スクロールバーや階層メニュー、TreeView、ComboBox などの特徴をすべて兼ね備えています。

LensBar は、普通のスクロールバーにフィルタリング用のテキスト入力領域を追加し、以下の機能を統合したものです。

リストのズーミング

スクロールする画面上で表示される項目の数は限られています。そこで、項目の重要度 (DOI：Degree Of Interests)[4] に応じてズーミング操作をおこない、表示する項目を制御して大きなリストを扱うようにしています。マウスを左右にドラッグしてズーミングレベルを変化させていくと、現在のズーミングレベルよりも大きい DOI 値をもつ項目だけが表示されます。

パターンマッチによる表示の間引き

LensBar では、ズーミングによる表示の間引きに加え、フィルタリング用のテキスト入力領域で指定された検索パターンにマッチする項目のみを表示する動的なフィルタリングもおこないます。これによって、表示される項目の数を減らします。つまり、grep のようなフィルタリング機能がつねに働いており、パターンにマッチしたデータだけが表示され、操作できるようになっているわけです。

動的曖昧検索

検索パターンにマッチする項目がない場合には、曖昧検索 (approximate pattern matching) が自動的におこなわれ、パターンにもっとも近い項目が表示されます [5]。

パターンマッチの結果の視覚化とブラウジング

パターンマッチの成功/失敗に応じて対応するスクロールバー上の位置の表示色を変更し、全データのなかでどの程度マッチする部分があるかを示します。さらに、スクロールバーでは、パターンマッチが成功した項目に対応する位置の上にノブが表示されるため、検索結果が効率的にブラウジングできます。

LensBar の使用例

英和辞書

LensBar を英和辞書に適用した例を図 3 に示します。画面上のテキスト・ウィンドウで検索パターンを指定すると、パターンマッチの結果がスライダの背景に表示されます。初期状態 (図 3-a) では検索パターンを指定していないため、すべての単語がマッチし、スライダの背景には矩形が表示されています。すべての単語のうち、スライダノブの中心の横線部分に対応した単語が右側に表示されています。この状態で左側のスクロールバーを動かしたり、スクロールバー領域をドラッグすれば (図 3-b)、すべての単語をブラウズすることができます。

動的曖昧検索をおこなうと、図 3-c〜d のように特定の文字列を含む単語だけを表示したり、マッチする単語がない場合に綴りがもっとも近い単語を探して表示することができます。曖昧検索が実行されているときは曖昧度に応じてパターンの背景が暗くなります。

この例では、検索にマッチした項目に対して DOI 値を図 3-e のように割り当てています。このように、選択項目 (この図では `graphics´) の位置と各項目の位置の差を 2 進数で表現したときの LSB (Least Significant Bit) の `0´の数を DOI 値とすれば、ズーミングレベルを連続的に変えたときに単語を半分ずつ間引いて表示していくことができます。

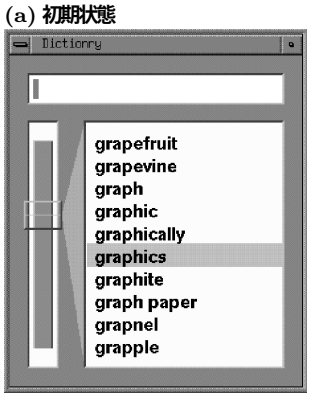
メールブラウザ

図 4 は、LensBar をメールブラウザに適用した例です。上側の LensBar でメールのタイトルのリストが表示され、選択されたメールの本体が下側に表示されています。下側の LensBar では検索パターンは指定されないので、通常のスクロール・ウィンドウとして動作しています。検索パターンとして `uist´を指定すると、本文に文字列 `uist´を含むメールだけが選択されます (図 4-b)。メールは時間順に並んでいるため、検索結果のパターンからこの件がアクティブだったおおよその時期を思い出すことができます。さらに、Lifestreams システム [6][1]のよ

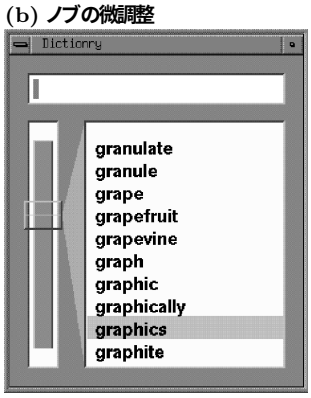
1 Lifestreams とは、ファイル操作などのアクティビティを時間軸だけで管理する `超整理法´的なアプローチです。

図 3　辞書への応用

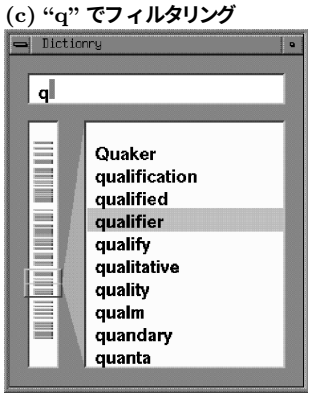
(a) 初期状態



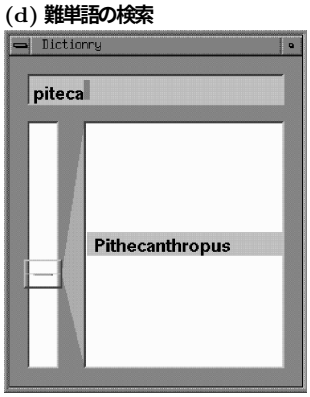
(b) ノブの微調整



(c) “q” でフィルタリング



(d) 難単語の検索



(e) DOI 値

DOI 値	エントリ
.....	.....
0	granule
1	grape
0	grapefruit
2	grapevine
0	graph
1	graphic
0	graphically
∞	graphics
0	graphite
1	graph paper
.....	.....

(f) 256 個ごとに表示

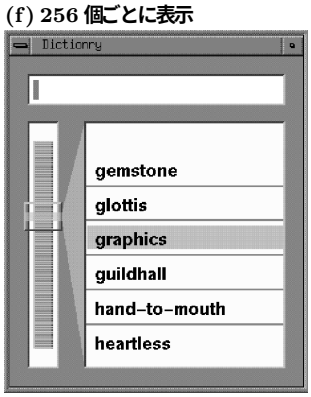
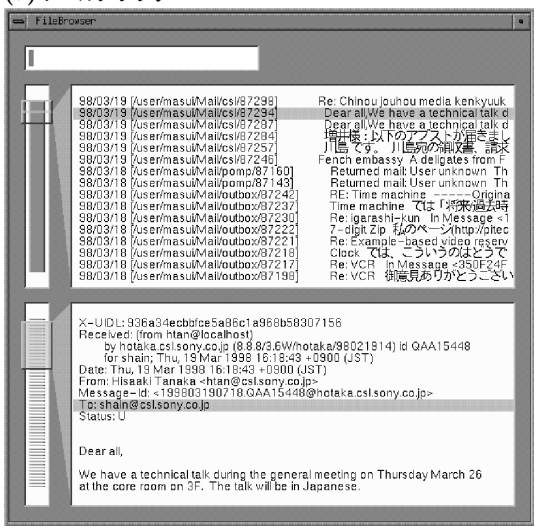


図 4　メールブラウザ

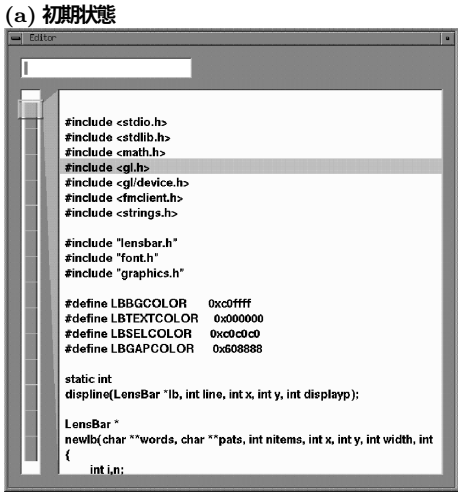
(a) メールブラウザ



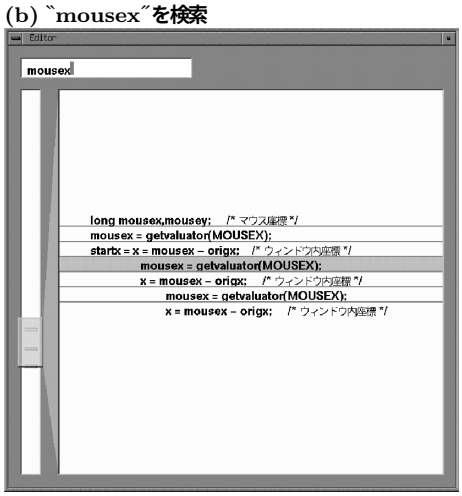
(b) `uist´でフィルタリング

図 5　プログラム・ブラウザ

(a) 初期状態



(b) `mousex´を検索



(c) 重要行も表示

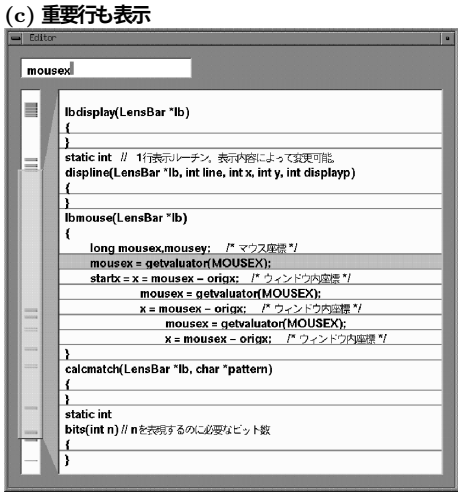


図 6　プログラムブラウザの DOI 値

DOI 値	エントリ
0	}
0	lbmouse(LensBar *lb)
0	{
5	long mousex,mousey;
−1	long origx,origy;
	.....
5	mousex = getvaluator(MOUSEX);
5	startx = x = mousex - origx;
	.....
3	mousex = getvaluator(MOUSEX);
	.....
2	mousex = getvaluator(MOUSEX);
−4	display();
−1	}
0	}

うに、時間的に関連のあるメールをブラウズすることができます。

プログラム・ブラウザ

図 5 は、LensBar をプログラムのブラウジングに適用した例です。プログラムリストの初期状態（図 5-a）から、`mousex´をキーとして検索すると図 5-b のようになります。マッチングの結果とその分布から、これは局所的に使われている変数であることが分かります。

この例では、DOI 値は図 6 のように割り当てています。マッチした行には正の DOI 値、失敗した行には負の DOI 値を割り当てますが、C のプログラムではインデントの小さい行が重要であることが多いため、インデントの小さな行ほど DOI 値が大きくなるようにしています。マウスを右にドラッグしてズーミングレベルを小さくすると、マッチしていない行も順に表示されるようになります（図 5-c）。この状態では、マッチした行だけでなく、プログラムの重要な行（関数宣言など）も表示されています。このように DOI 値を割り当てることによって、Focus + Context 機能を実現しています。

階層メニュー

UNIX の /usr ディレクトリを、階層メニューとして LensBar で表示した例を図 7 に示します。ズーミングレ

図 7　階層メニュー

(a) 階層メニュー

(b) サブディレクトリを拡大

図 8　全体表示

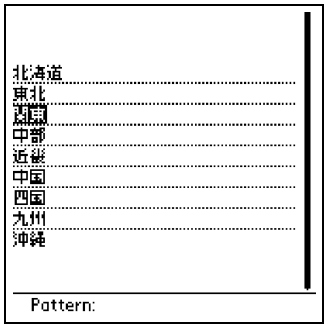


図 9　関東地方にズームイン

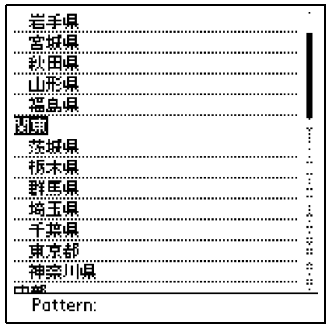


図 10　茨城県にズームイン

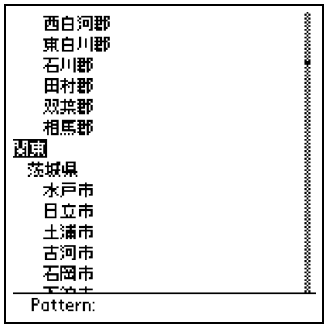


図 11 `no´を含む地名をフィルタリング

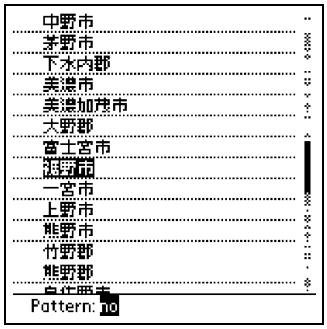


図 12 `noh´を含む地名のリスト

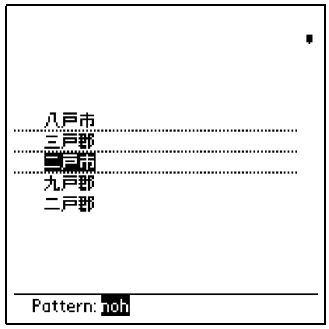
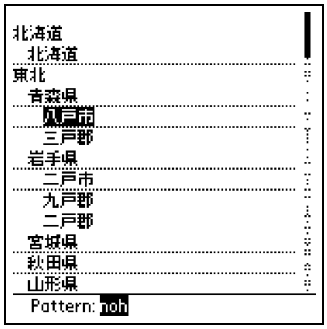


図 13 `noh´を含む地名の周辺情報表示



ベルを大きくすると上位階層のみが表示され（図 7-a）、小さくすると深い階層まで表示することができます（図 7-b）。通常の階層メニューとは異なり、a と b の状態は連続的/可逆的に遷移可能なのが特徴です。

PalmPilot 上の LensBar

LensBar はもちろんパーソナル・コンピュータの GUI 画面でも有効ですが、PDA のような `Baby Face´端末においてとくに大きな効果を発揮します。

図 8 は、PalmPilot 上の LensBar で日本の地名を表示しているところです。`関東´をタップしてペンを右に動かすと県がリストされ（図 9）、さらに右に動かすと市や郡が表示されます（図 10）。

検索パターンを入力すると、それにマッチする地名だけがリストされます。たとえば `no´と入力すると、`の´という読みを含む地名だけがリストされ（図 11）、さらに `h´を加えて指定すると `noh´を含む `八戸´などの

図 14　予定表の初期画面

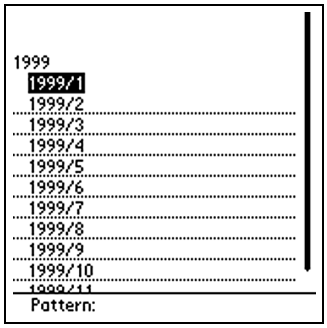


図 15　`uis´でフィルタリング

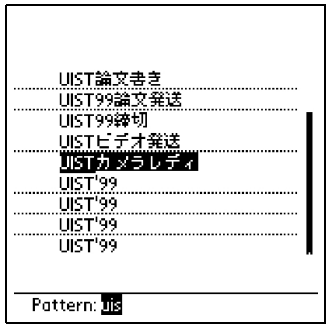
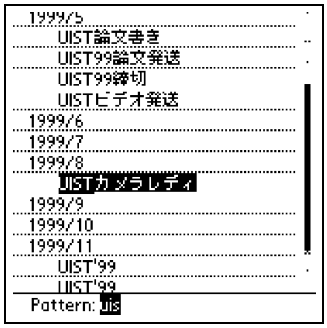


図 16　ズームイン表示



地名だけが表示されます（図 12）。

ここで、ペンで `八戸´を選択して右に動かすと、読みに `noh´が含まれる八戸市などがある `岩手´や `青森´といった県名が表示されます（図 13）。

■ 表現をすこし変更しました。

図 14 は、LensBar でスケジュール・データをブラウジングしているところです。まず、パターンとして `uis´を指定してフィルタリングをおこないます（図 15）。続けて、その周囲のデータを表示することにより、UIST カンファレンス関連の仕事をいつごろおこなっていたかをブラウズすることができます（図 16）。

ここで扱っているデータは、1,000 行を超えています。したがって、通常のスクロールバーなどでは、全体を効率よくブラウジングすることは困難です。このような場合にも、LensBar でズーミングやフィルタリングを併用すれば、小さな画面でも検索やブラウジングが効果的におこなえるようになります。

おわりに

今回は、大量の情報のブラウジングとフィルタリングの両方をサポートする GUI 部品として `LensBar´を紹介しました。膨大な情報が流通している現代社会では、多種多様な情報をフィルタリングしたのちに眺めたり、編集したりといった手法がますます重要になってくるでしょう。

LensBar は、メニューやスクロールバーなどの基本的なインターフェイスの上位互換ツールと捉えることができます。したがって、 比較的単純なデータを扱う場合には有用ですが、今後はさらに複雑で大規模なデータにも対応できるブラウジングやフィルタリングのための基本ツールが必要になってくるでしょう。

■ プログラムの入手先は?

（ますい・としゆき　ソニー CSL）

[参考文献]

[1] Manojit Sarkar and Marc H. Brown, “Graphical fisheye views”, *Communications of the ACM*, Vol. 37, No. 12, pp. 73–83, December 1994

[2] Toshiyuki Masui, “LensBar–visualization for browsing and filtering large lists of data”, In *Proceedings of IEEE Symposium on Information Visualization (InfoVis'98)*, pp. 113–120, October 1998

[3] 増井俊之「ブラウジングとキーワード検索を統合した GUI 部品 LensBar」、安村通晃（編）『インタラクティブシステムとソフトウェア VI：日本ソフトウェア科学会 WISS'98』、pp.153–158、近代科学社、1998 年 12 月

[4] George. W. Furnas, “Generalized fisheye views”, In *Proceedings of the CHI'86 Conference on Human Factors in Computing Systems and Graphic Interfaces*, pp.16–23, Addison-Wesley, May 1986

[5] 増井俊之、水口 充、George Borden、柏木宏一「なめらかなユーザインタフェース」、第 37 回冬のプログラミングシンポジウム予稿集、pp. 13–23、情報処理学会、1996 年 1 月

[6] Eric Freeman and Scott Fertig, “Lifestreams: Organizing your electronic life”, In *AAAI Fall Symposium: AI Applications in Knowledge Navigation and Retrieval*, November 1995