情 報 学 基 礎　66－3
デジタル・ドキュメント　32－3
（2002．3．15）

時間軸を考慮した知的情報配信技術

宮下一博† 後藤文太朗‡
北見工業大学 情報システム工学科
miyakz@pattern.cs.kitami-it.ac.jp† goto@cs.kitami-it.ac.jp‡

電子メールを使った情報配信に関して、単に受信者の興味により電子メールの振り分けを行うだけでなく、受信者の都合のいい時間帯、発信者にとって効果的な時間帯といった、時間軸を考慮した情報配信を行うシステムの設計について述べる。本システムのTime Filterにより、情報の受信者に対する生活時間帯への配慮が可能になり、また、情報発信者に関しては、時間帯指定を行うことによる、受信者の絞り込みが実現できた。

Intelligent Information Delivery Methods using Temporal Information

Kazuhiro MIYASHITA† , Fumitaro GOTO‡
Department of Computer Sciences, Kitami Institute of Technology
miyakz@pattern.cs.kitami-it.ac.jp† goto@cs.kitami-it.ac.jp‡

We introduce temporal information into an information delivery system. The temporal information introduced are sender's history of publishing, receiver's acceptable time span of E-mail, article's time span that is effective. So we make core design of the system and show that the system enables us individual attention of receivers about his time. Additionally, the system enables us effective publishing for sender.

1. はじめに

近年、インターネットにおける情報洪水のため、インターネットから必要な情報を得たり、インターネット上へ効果的に情報を配信したりすることが困難になってきている。

その解決策として、ユーザのアクセス履歴を活用したブラウジング支援[5][9]や、サイト管理技術の研究[6]が行われ、電子メール利用に関しては、フィルタリングによる情報獲得支援[7]や、知識ベースを用いた情報配信支援に関する研究[8]が行われてきた。また、WWWを用いた情報配信において、ユーザの興味属性を自動的に把握し、WWWサーバ側で保持する情報群からユーザの興味に応じて適切な情報を紹介する方式も提案されている[3]。

一方、本来の目的である、商売等の目的達成の工程において、多くの場合、時間的要因は非常に重要なものであり、時間粒度に関する研究も行われている[2]。そこで我々は、時間軸を考慮した知的情報配信技術を提案する。

本稿で提案する知的情報配信技術は、電子メールを使った情報配信に関して、以下のような事柄を可能にする。

- 情報受信者の生活時間帯への配慮
- 情報発信者に関して、発信する情報（広告など）に時間帯指定を行うことによる、受信者の絞り込み効果

2章以降、上述のシステムの設計について説明を行う。最初に、仲介システムが存在する情報配信システムの共通点を抜き出し、モデリングを行う。次に、モデル内に存在する各オブジェクトに対して、時間軸を考慮したモデリングを試みる。そして、実際の情報配信処理である、フィルタリングについて説明を行う。最後に北海道北見市の

地域情報データベースK-MINT[1]に本システムを適用した時の設計について述べる。

2. 時間軸を考慮した情報配信

2.1 Mediator 情報配信アーキテクチャ

情報配信の枠組みとして、本システムで採用している Mediator 情報配信アーキテクチャについて説明する。

メールマガジン代理配信システムは、マガジン作者個人に替わって、読者の管理や配信処理を行うシステムである。特に、個人のコンピュータの性能では配信できないほど読者を抱えている場合でも、システムに任せることで配信が可能になっている。概念図を図 2-1 に示す。

似たようなものに、メーリングリストのシステムなどがある。このようなシステムの共通点は、情報の発信者と受信者が複数存在し、1つの仲介システムが機能を持っていることである。

本稿では、このようなアーキテクチャを Mediator 情報配信アーキテクチャと呼ぶ。図 2-2 に概念図を示す。

今後、時間軸を考慮した情報配信を議論するにあたって、Mediator アーキテクチャをベースにして進めてゆく。そのため、図 2-2 中の各オブジェクトに関して、以下のような定義をする。

定義１：Sender
広告メールなどの情報を発信するもの

定義２：Article
Sender が発信する、上述のような情報

定義３：Receiver
Article を受信するもの

定義４：Mediator
Receiver に対して配信する Article を決定するもの

定義５：Deliverer
具体的な配信手段メールサーバなど

以降、時間軸を考慮した情報配信のための Receiver と Sender および Article、そして時間に対して一般的なモデリングを行ってゆく。表 2-1 に、広告配信と Mediator アーキテクチャとのマッピングの例を示す。

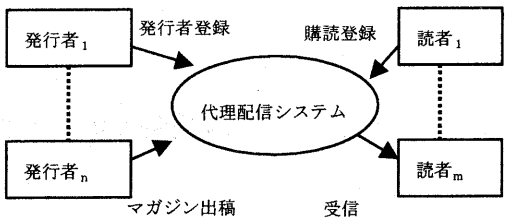

図 2-1：メールマガジンの代理配送システム

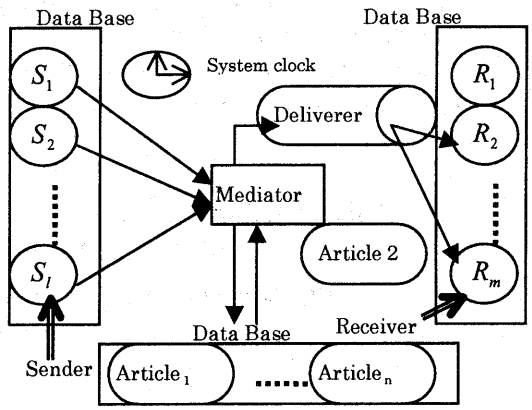

図 2-2：Mediator 情報配信アーキテクチャ

Mediator アーキテクチャ	広告配信
Sender	広告主
Receiver	受信者
Article	広告
Mediator	配信システム
Deliverer	メールサーバ

表 2-1：Mediator アーキテクチャのメーリングリスト

2.2. Sender および Article のモデル

Article に関する属性を以下のように定める。

1. タイトル（テキスト）
2. 内容（テキスト）
3. 「Receiver に受け取って欲しい時間帯と興味属性」
4. 送信元の Sender に関する情報

また、Sender は、属性として「連絡先メールアドレス」のみを持つものとする。

2.3. Receiver のモデル

Receiver に関する属性を以下のように定める。

1. メールアドレス
2. 「メールの受信希望時間帯と興味属性」というペアのリスト

さらに、受信希望時間帯については、「メールを受け取る上限回数 f と、メール一通あたりに含まれる Article の上限数 g 」の２つのパラメタを付加する。

2.4. 時間のモデリング

柔軟な時間帯指定を可能にするような時間の粒度に関するモデリングを行う。

図 2-3 に関連クラスの UML 図を示す。図中、始点が黒菱形の矢印は集約関係を表し、他方は継承関係を表す。

2.4.1. 原始的な要素

クラス：Instant

Instant クラスは、基準時からのミリ秒を整数として表したものである。また、基準時を GMT 標準時と定める(１９７０年1月１日午前０時)。

これは、時間に関する最小要素である。以下にこのクラスの定義を、Java 言語を簡略化したもので示す。なお、アクセサやコンストラクタの類は一部省略する場合がある。

```
public class Instant {
        private long msecFromGMT;
        //アクセサ、コンストラクタの類は省略
}
```

クラス：Time Span

Instant は時間軸上の一点であったが、TimeSpan クラスは、時間軸上のある範囲を表現するクラスである。始点と終点は Instant オブジェクトで表される。以下にクラス定義の一部抜粋を示す。

```
public class TimeSpan {
        private Instant start , end;
```

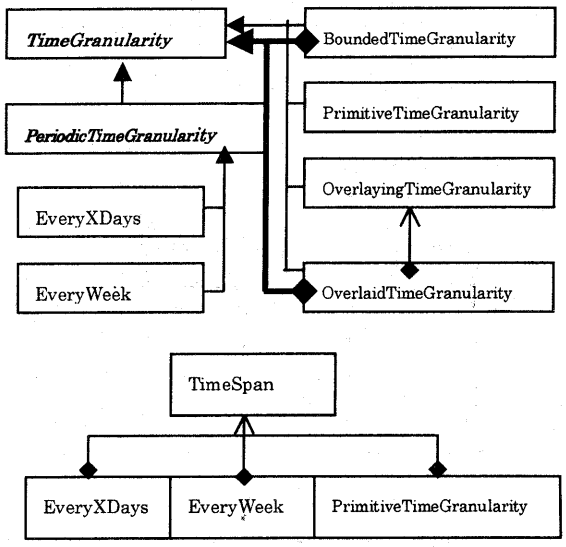

図 2-3：時間粒度クラス群の UML 図

```
        boolean isInclude( Instant i )
        //アクセサ、コンストラクタの類は省略
}
```

isInclude メソッドは、時間帯に Instant オブジェクト i を含む（start≦i≦end）とき、true を返す。

2.4.2. 時間の粒度

クラス：Time Granularity

次に、時間粒度を表現する TimeGranularity クラスを定義する（以下、TimeGranularity という語を TG と略す場合がある)。

このクラスは、TimeSpan オブジェクトの集合を管理する。また、図 2-3 にあるように、クラス階層の頂点であり、抽象クラスである。以下に TG クラスの完全な定義を示す。

```
public abstract class TimeGranularity {
abstract public boolean isInclude( Instant);
abstract public TimeSpan getTimeSpan(Instant i);
}
```

このクラスの isInclude メソッドが true を返すときは、isInclude が true を返すような TimeSpan が少なくとも１つあるときである。

また、getTimeSpan ではＴＧの isInclude

が true であるとき、該当する TimeSpan オブジェクトを返す。

図 2-4 にイメージを示す。

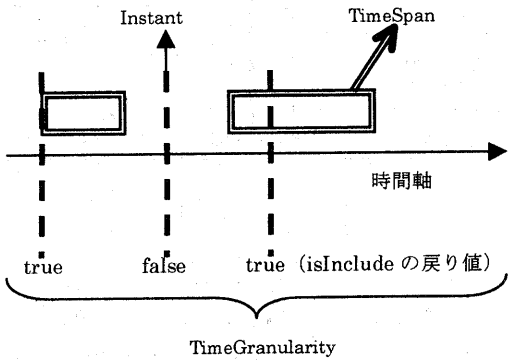

図 2-4：Time Granularity と Time Span

クラス：PrimitiveTimeGranularity

このクラスは「ある１日」を表す時間粒度である。フィールドに TimeSpan オブジェクトを１つだけ持ち、その始点と終点は、ある１日中に収まるように限定している。

この基本単位を組み合わせて、複雑な時間帯を形成できるように、以下のモデリングを行う。

2.4.3. 周期的・境界付け

クラス：PeriodicalTimeGranularity

このクラスは、周期的に繰り返す Primitive TG を表現する抽象クラスである。フィールドに原点として、Primitive TG オブジェクトを持つ。

このクラスの直系に、「X 日毎」を表す EveryXDay や「毎週 Y 曜日」を表す EveryWeek などがある。

クラス：BoundedTimeGranularity

このクラスは、境界付けをしたい TG オブジェクトと、境界の始点と終点を表す２つの Instant オブジェクトをフィールドに持つ。

2.4.4. 重ね合わせによる時間帯の合成

以下では、TimeGranularity を重ね合わせることで、複雑な時間帯を形成する仕組みを提供する２つのクラスについて述べる。

クラス： OverlayingTimeGranularity

重ね合わせの単位である。フィールドに TG オブジェクトを持ち、重ねた部分を消去するか否かを決める属性を付加する。以下に、クラス定義の抜粋を示す。

```
public class OverlayingGranularity
extends TimeGranularity
{       private boolean erase;
        public boolean isErase(){return erase;}
        public setErase(boolean b){erase = b;}
}
```

setErase メソッドで消去属性の設定を行う。引数に true を与えれば、重ねた部分を消去する属性になる。また、isErase メソッドで属性の確認を行う。

クラス： OverlaidTimeGranularity

このクラスは、OverlayingTimeGranulariry をスタック状に管理し、時間帯の重ね合わせを行う。

スタックの最初の要素は初期値として与えられる。クラスの定義の抜粋を以下に示す。

```
public class OverlaidGranularity
extends TimeGranularity
{       private Stack overlayStack;
        public OverlaidTG(OverlayingTG..);
        public add(OverlayingTG..);
        public remove(OverlayingTG..);
}
```

以後、要素を操作するために追加された add メソッドと remove メソッドを使って、時間帯を重ね合わせてゆく。

このクラスは TimeGranularity を継承するので、isInclude と getTimeSpan メソッドを持つが、その動作が特徴的なので以下に説明する。

１）isInclude メソッドの動作

このメソッドはスタックの頂点から順に要素を検査してゆき、以下の条件で返り値を決める。

（条件１）

OverlayingTimeGranularity の isInclude メソッドの返り値が true かつ isErase が false ならば、OverlaidTimeGranularity の isInclude は true を返す。つまり、時間帯が上書きされたことになる。

（条件２）
OverlayingTimeGranularity の isInclude メソッドの返り値が true かつ isErase が true ならば、OverlaidTimeGranularity の isInclude は false を返す。つまり、時間帯が一部分消去されたことになる。

２）getTimeSpan メソッドの動作

isInclude 同様、スタック中の要素を最初から順に検査してゆき、以下の条件で返り値を決める。

（条件１）
OverlayingTimeGranularity の isInclude メソッドの返り値が true かつ isErase が false ならば、該当要素が持つ TimeSpan オブジェクトを返す。

（条件２）
OverlayingTimeGranularity の isInclude メソッドの返り値が true かつ isErase が true ならば、getTimeSpan は無効な値（null）を返す。

2.5. 興味属性

本研究における興味属性は、その Article が持つ「カテゴリ」を採用する。配信される Article へのカテゴリは Sender のカテゴリを継承するものとする。これは、Sender が属するカテゴリが「食料品」なら、それに准じた Article を発信するだろうという根拠に基づいたものである。

2.6. Mediator 部の詳細化

時間帯や興味属性（カテゴリ）のフィルタリング処理を実現する Mediator を図 2-5 に示す。

現段階のシステムでは、興味属性に関してフィルタリングを行う InterestFilter と、時間帯に関してフィルタリングを行う TimeFilter がある。図 2-6 中では、InterestFilter の処理結果が TimeFilter に渡っている。これは、まず興味合った Article を決定した後で、時間帯に合うものを選び出すという事を示している。

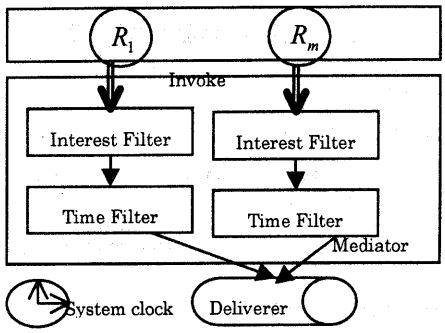

図 2-5：各 Filter と Receiver

また、各 Receiver は自分自身のフィルタのリストを持っている。これは将来的にシステムが取り扱うフィルタの種類が増えた場合に、使用するアルゴリズムの交換を可能にするための措置である。

システムクロックは、情報配信システムシステムの動作のタイミングを定めている。

以下、InterestFileter と TimeFilter の働きについて説明する。

2.6.1. Interest Filter

Article のカテゴリと、Receiver の受け取り時間帯に付加された、興味属性（カテゴリ）のマッチングを行う。カテゴリは具体的には整数で表され、変数に保持される。マッチングは変数同士の比較という簡便な方法で行われる。カテゴリがマッチした広告をリストに格納して出力し、次の TimeFilter に引き渡す。

2.6.2. Time Filter

時間帯マッチングによるフィルタリングアルゴリズムの概略を述べる。図 2-6 に概念図を示す。

時間帯がマッチするとは、Receiver と Article の時間帯が重なることを言う。よって、図中において、Receiver は３通のメールを受け取る。

時間帯がマッチしたら単に配信するという方式でも、Article の数が少ない内はまだ良いが増えてくるとそうは行かなくなる。例えば、本システムの Receiver は、携帯端末の使用を想定しているが、時間帯にマッチする１００個の Article がメールで配信されたならば、それだけの回数分、

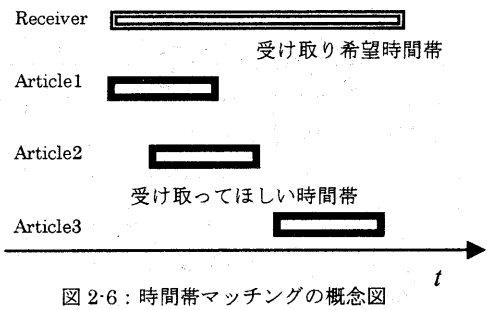

図 2-6：時間帯マッチングの概念図

着信ベルが鳴るのである。これでは非常に煩わしい。

このフィルタのアルゴリズムでは、Receiver の受け取り時間帯に付加されている、メールの受信上限回数 f と、一回の配信に含まれる Article の上限 g というパラメタを使うことにより、前述のような問題の軽減を試みている。

フィルタが起動されると、受け取り時間帯に付加された f と g を満たすように時間帯のマッチングを行う。例えば、図 2-6 中の受け取り希望時間帯に f=2、g=2 が設定されていたなら、前半に 2 つの広告をまとめて一回、間隔を空けて後半に配信するといった処理を行う。間隔は時間帯を f で割った値を用いる。

なお、時間帯にマッチしなかった場合や、f や g の許容範囲を超えてしまった場合、該当する Article は配信されないようにする。

3. K-MINT との統合

3.1. システム統合と全体の動作の流れ

北海道北見市の地域情報データベース K-MINT[1]には、市内の様々な情報が格納されている。６００以上の広告主が登録されており、約４０種のカテゴリに分類されている。また、キーワード検索やクーポンなどのサービスが利用できる。

WWW に公開されており、携帯電話での閲覧を想定した設計になっているが、PC でも問題なくアクセスすることができる。

ここでは、K-MINT の既存の機能を温存しつつ、時間軸を考慮した情報配信機能を追加し、K-MINT の機能強化を試みる。図 3-1 にシステム統合図を示す。

以下、システム全体の実行の流れを図中の各 Step に沿って説明する。

- **Step 1 ：**
  広告メール受信者（Receiver）は、本システムに受け取り希望時間帯や興味属性などの属性を登録する。また、広告主（Sender）については、K-MINT のデータベースにすでに登録されている場合もあるが、新規登録もできる。

- **Step 2 ：**
  広告主は、システムを通じて広告（Article）を配信する。その時に、広告にも同様に属性が付加される。

- **Step 3 ：**
  各受信者に対して配信する広告を決定する。情報配信システム（Mediator）によって、時間帯および興味属性のフィルタリングが行われる。

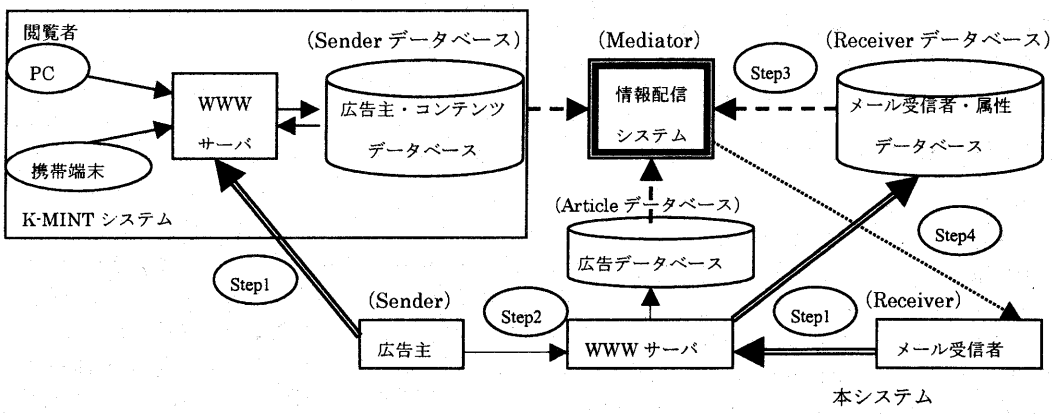

図 3-1：時間軸を考慮した情報配信機能を追加した K-MINT

```
1:  efri = new EveryWeek( "Fri" , 17,21);
2:  bfri = new BoundedTG( efri , start , end);
3:  ov1 = new OverlayingTG(bfri);
4:  ov1.erase(false);
5:  pri = new PrimitiveTG( 2002,1,13);//臨時休業日
6:  ov2 = new OverlayingTG( pri );
7:  ov2.erase( true );
8:  result = OverlaidTG( ov1 );
9:  result.add( ov2 );
```

図 3-2：時間オブジェクトによる時間の組み立て

- **Step 4 ：**
  Mediator がメールサーバ（Deliverer）を通じて広告メールを配信する。

3.2. 時間オブジェクトによる時間の組み立て

3.2.1. 例題

K-MINT の広告主を調査すると、夜間営業の飲食店が１６％と最多であった。そのような広告主の場合、メールを用いた効果的な広告配信の例として、次のようなものが考えられる。以下、これを例題と呼ぶ。

『毎週金曜日の午後 7 時～９時の時間限定で、広告メールを配信したい。メール内のインタラクションURLをクリックしてくれた限定 15 人にクーポンをメールで配信する。しかし、臨時休業日が１日だけあり、それがちょうど金曜日であるので、この日だけは広告を出したくない。そして、このサービス期間は７月１日から翌月１日までの期間限定である。』

以下、この例題中に現れる時間帯を、第２章で定義した時間のモデルで表現する。

3.2.2. 実践

3.2.1 の例題の、広告を出す時間帯を構成するための擬似コードの断片を図 3-2 に示す。以下では、図中のコードについて説明を行う。

１行目：
毎週金曜日の午後７時～９時という EveryWeek オブジェクトを作る。

２行目：
また、期間限定なので、これを BoundedTG にする。なお、境界の始点は７月１日午前０時０分の Instant、終点も同様に指定する。図中では start と end と簡略化されている。

３、４行目：
後で重ね合わせを行いたいので、これを OverlayingTG にする。そして消去属性を false にする。

５、６、７行目：
ここで、臨時休業日が１日だけあるので、それを PrimitiveTG で表現する。同様に OverlayingTG 化し、消去属性を true にする。

８行目：
重ね合わせによる時間帯を作りたいので、ここで OverlaidTG を作る。

９行目：
休業日を表す、図 3-2 中の ov2 を重ね合わせる。

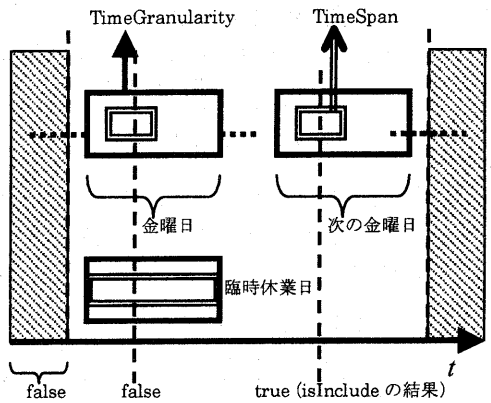

図 3-3：例題における時間帯のモデリング

結果として出来た新しい TG オブジェクトのイメージを図 3-3 に示す。

4. おわりに

時間軸を考慮した情報配信のための基礎作りを行った。時間帯に関するモデリングを行い、3.2.1 の例題のような柔軟な時間帯指定が可能になった。

既存の K-MINT の機能に、以上の概念を導入した情報配信が可能になるシステムを設計した。

これにより、以下の事柄が可能になった。

1）情報受信者の生活時間帯への配慮

Time Filter により、自動的に Receiver の希望時間帯にメールが届くようになることから、これが実現できた。

2）受信者の絞り込み効果

Time Filter の処理により、Article を受け取って欲しい時間帯と、Receiver の受け取り希望時間帯のマッチングが行われ、自動的に Receiver を絞り込めるようになった。

しかし、現状では以下の点において問題が残っている。

1）Time Filter

Time Filter のアルゴリズムは、パラメタ f と g をあまりにも忠実に守るため、例えば、Receiver が本当は欲しかった Article が手に入らなくなるといった可能性が生じる。

2）ユーザビリティ

Receiver の生活時間帯は常に一定とは限らない。変わった場合、現状のシステムでは再び時間帯を手動で入力しなければいけないため、ユーザビリティが低いという問題点がある。

今後は、これらの点の解決および、Java 言語によるシステムの実装を進めていく。

参考文献

[1] 地域情報データベース K-MINT : http://www.0157.net

[2] Claudio Bettini, Sushil Jajodia, Xiaoyang Sean Wang : "Time Granularities in Databases, Data Mining, and Temporal Reasoning"

[3] 橘高博行, 佐藤直之, 鈴木英明, 曽根岡昭直 : "パーソナライズ情報提供方式の提案と評価", 情報処理学会論文誌, Vol.40 No.1

[4] 長谷川隆明, 浅野久子, 堀井統之 : "電子メールのインテリジェントサービス", 人工知能学会誌, Vol.14 No.6

[5] 石川雅弘,後藤文太朗 : "WWW アクセス活動と Web コンテンツの情報統合", 情報処理学会「第60回全国大会」講演論文集, pp.157-158,2000

[6] 上野貢,後藤文太朗 : "Web の構成要素の部品化とその合成による Web サイト構築",情報処理学会第62回全国大会,2001

[7] 長谷川隆明,高木伸一郎 : "文書構造の認識と言語の特徴の利用に基づく電子メールからのスケジュールと ToDo の抽出", 情報処理学会論文誌, Vol.40 No.10

[8] 上田宏高,門林理恵子,萩野浩明,塚本昌彦,西尾章治郎 : "知識ベースシステムを用いた分散型メール配送システム MILD", 情報処理学会論文誌, Vol.39 No.2

[9] Henry Lieberman : "Letizia ; An Agent That Assists Web Browsing",http://lieber.www.media.mit.edu/people/lieber/Lieberary/Letizia/Letizia-AAAI/Letizia.html