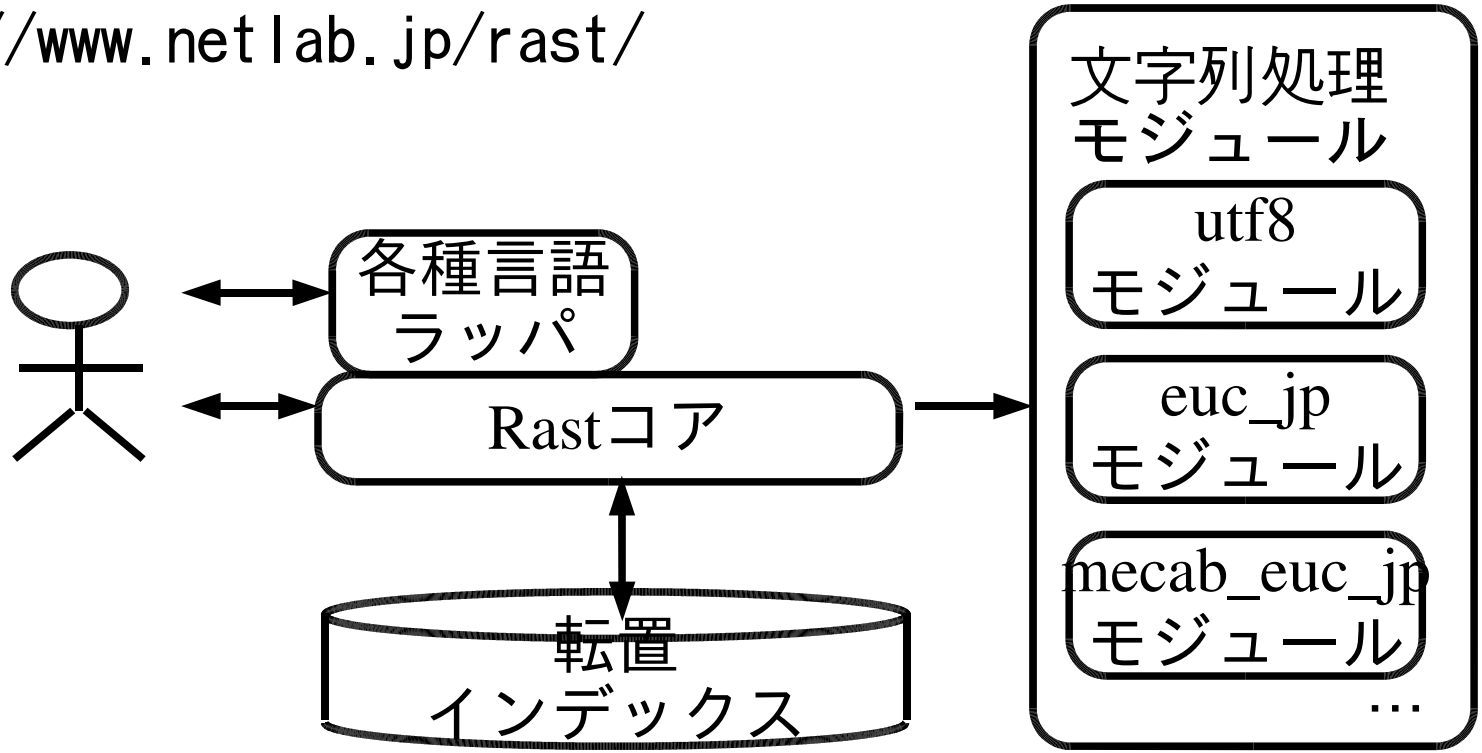
全文検索システムRastの設計と実装

株式会社 ネットワーク応用通信研究所 雄也

Rastとは

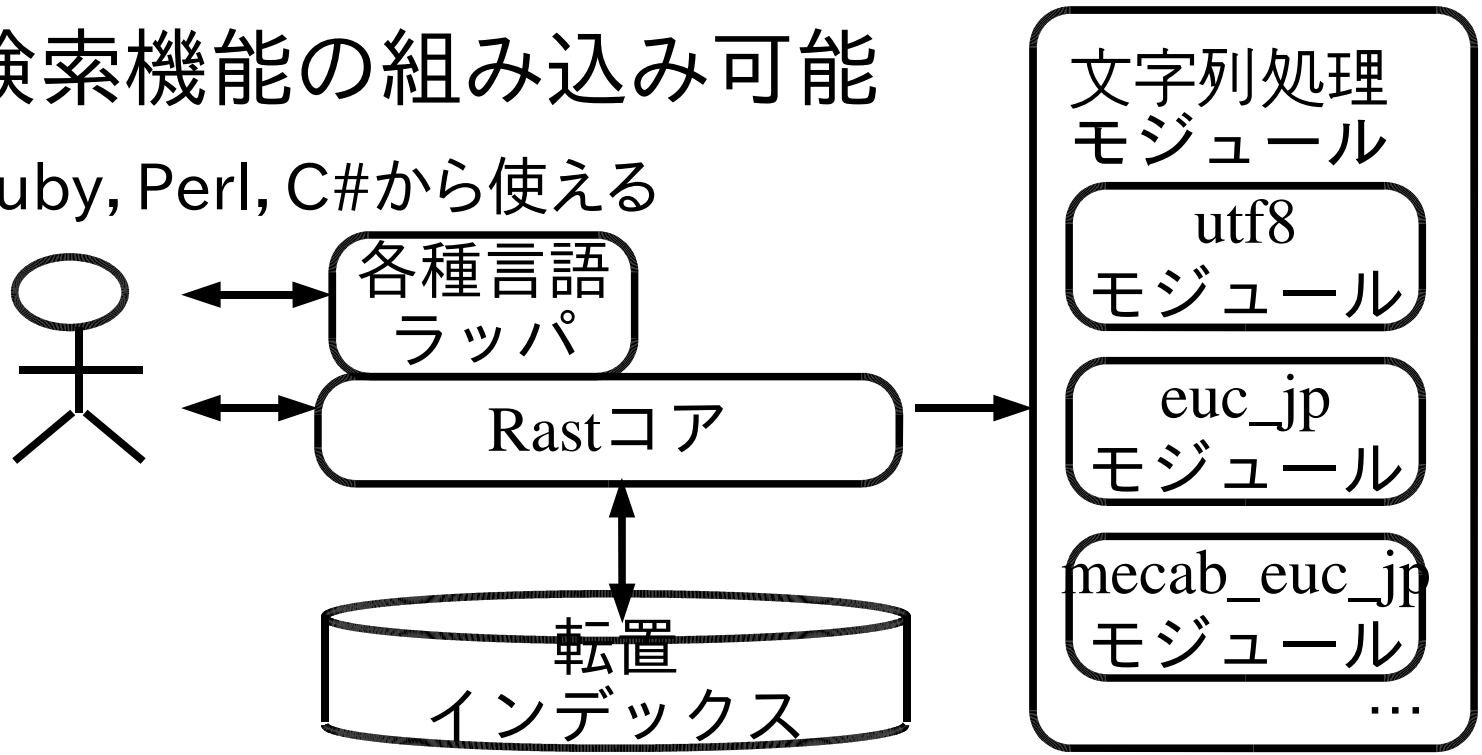
- 転置インデックスを利用する全文検索システム
- ハッカーに全文検索の自由をもたらす
- 自由ソフトウェア

http://www.netlab.jp/rast/



特徴

- 文字列処理モジュール
  - 検索方式が選べる（N-gram方式, 分かち書き方式）
  - 使用する文字エンコーディングが選べる
  - 検索や要約における正規化方式が選べる
- 全文検索機能の組み込み可能
  - C, Ruby, Perl, C#から使える



特徴

- 文字列処理モジュール
  - 検索方式が選べる（N-gram方式, 分かち書き方式）
  - 使用する文字エンコーディングが選べる
  - 検索や要約における正規化方式が選べる
- 全文検索機能の組み込み可能
  - C, Ruby, Perl, C#から使える
- プロパティ検索
- インクリメンタルな更新
- メタ検索

差し替え可能な文字列処理モジュール 機能

- N-gram方式や分かち書き方式といった検索方式の切り替えと 任意の文字エンコーディングでの転置インデックスの作成, 検索するために 以下の機能を差し替え可能な別モジュールにした.
  - 文字単位の情報取得
    - is_spaceなど
  - 索引語の切り出し単位の決定
  - 正規化
    - 要約表示用の正規化
    - 検索用の正規化

差し替え可能な文字列処理モジュール 効果

- モジュールの追加をする事で任意の文字エンコーディングに対応できる
- 文字列処理モジュールを切り換えれば，以下の用途にもRastが使用可能になる
  - 文字単位を1バイト固定にすることで，バイナリ対応

文字列処理モジュール utf8モジュール

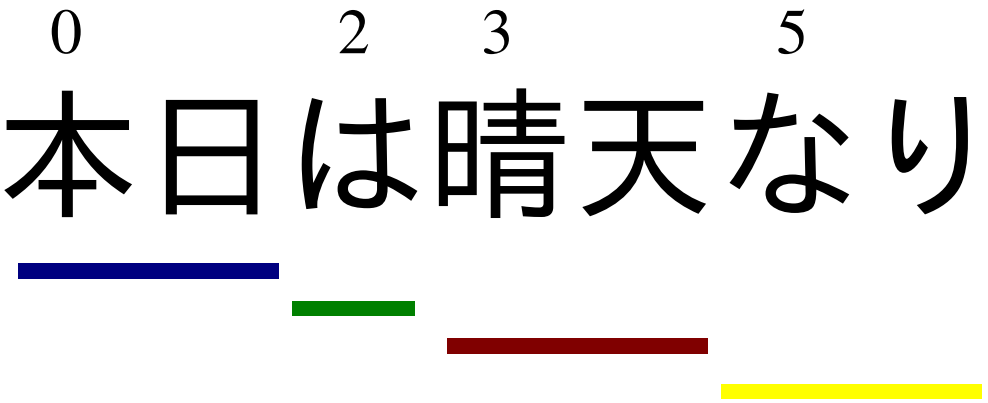
- 扱う文字エンコーディングはUTF-8
- 索引語の切り出しはN-gram方式
- N可変方式を使う（文字種によってNの値を決める）
  - ラテン文字とギリシア文字, キリル文字, 平仮名, 片仮名はtri-gram.
  - それ以外はbi-gram
  - 文字種の切り替えが発生したときはbi-gram
  - 文字種の判別はICU（IBMのUnicodeを扱うライブラリ）に任せている

文字列処理モジュール euc_jpモジュール

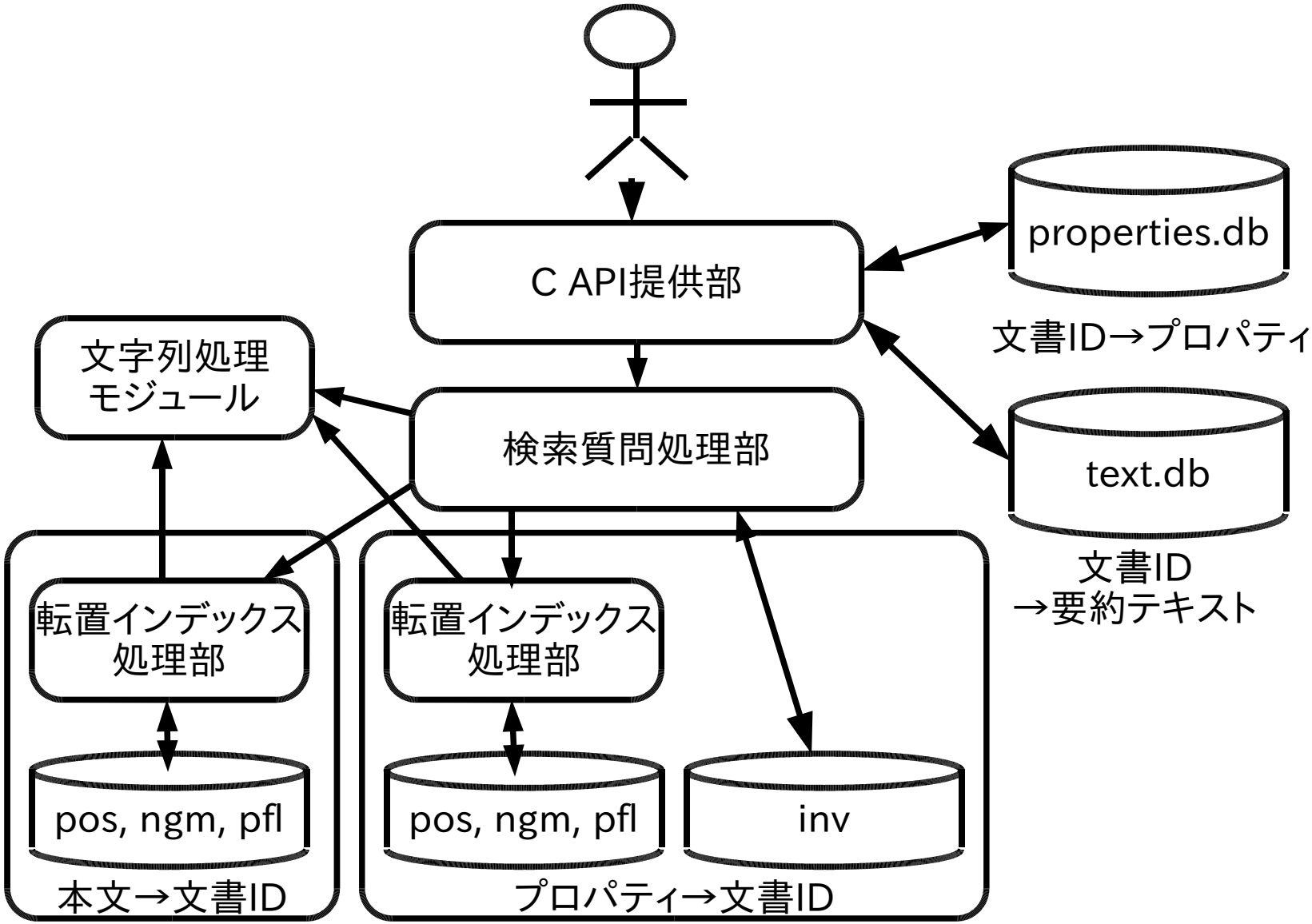
- 扱う文字エンコーディングはEUC-JP
- 索引語の切り出しはN可変のN-gram方式
  - 英数字文字と平仮名, 片仮名はtri-gram.
  - それ以外はbi-gram
  - 文字種の切り替えが発生したときはbi-gram

文字列処理モジュール mecab_euc_jpモジュール

- 扱う文字エンコーディングはEUC-JP
- 索引語の切り出しはMeCabを利用した分かち書き方式



検索処理 概要図


C API提供部
properties.db
文書ID→プロパティ
文字列処理
モジュール
検索質問処理部
text.db
文書ID
→要約テキスト
転置インデックス
処理部
転置インデックス
処理部
pos, ngm, pfl
pos, ngm, pfl
inv
本文→文書ID
プロパティ→文書ID

検索 overview

1)文字列処理モジュールを用いて索引語の切り出し

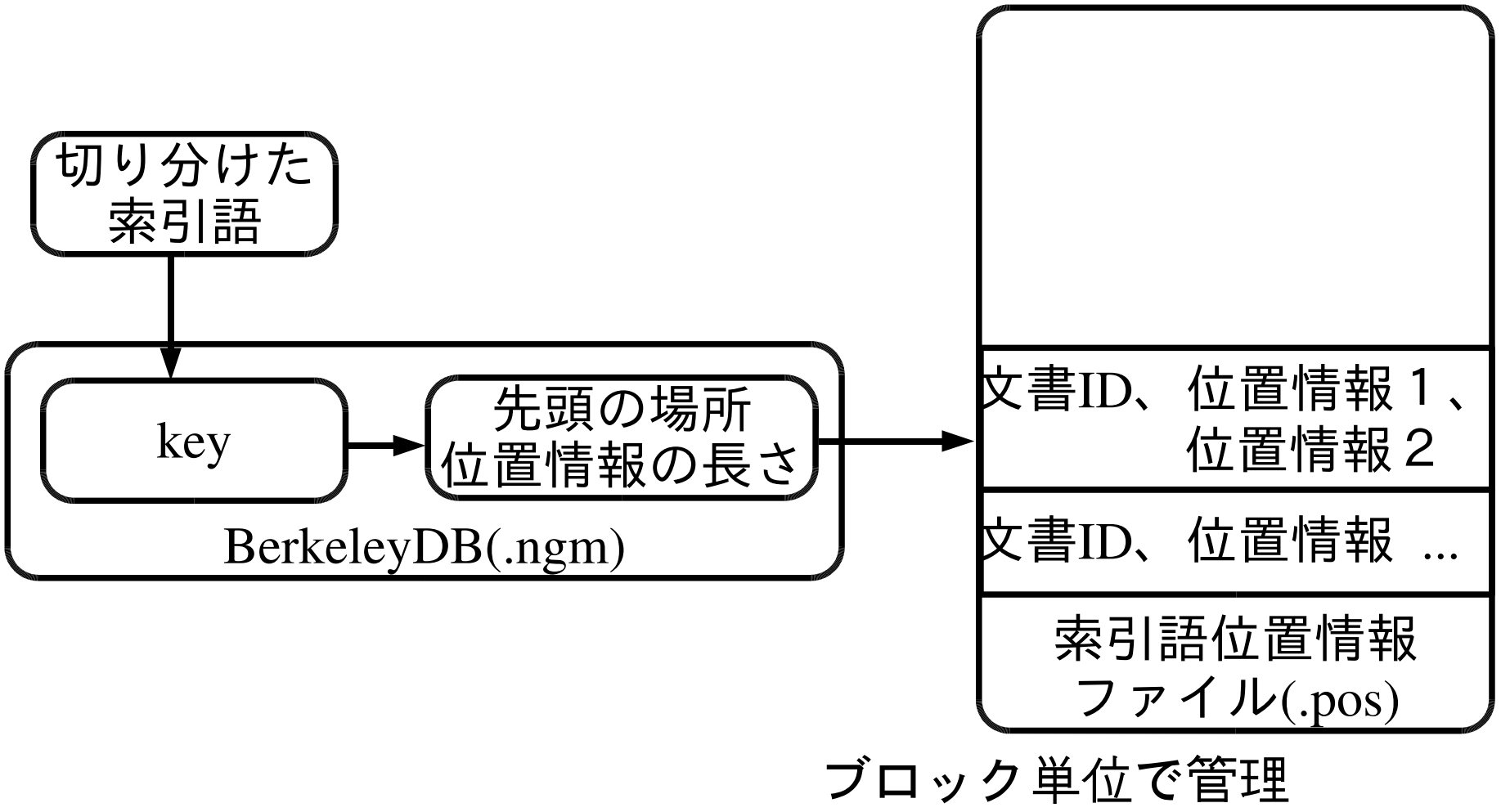
2)索引語をキーにB木（Berkeley DB）に格納されている先頭の場所, 領域の長さを検索

3)文書中の位置情報が格納されているファイルから文書IDと位置情報を読み込む

4)隣接判定を行う

5)結果を返す

位置情報ファイル構造図



検索 位置情報の管理方法

- 文書中の位置情報の管理
  - 索引語ごとの文書中の位置情報を一定のサイズのブロックで管理
  - 先頭の場所（block_no）とその数（block_count）は索引語をキーにBerkeley DBで管理
  - 1ブロックのサイズはデフォルトで512バイト（変更可能）

検索 低頻度な索引語

- 位置情報ファイルでの未使用領域の調査
  - 出現数の少ない索引語が大半
- 稀にしか出現しない索引語の管理方法を特別に扱う
  - 対象文書群全体で1箇所にしか出現しない索引語を分けて管理
  - 1要素のサイズが大きくないので直接Berkeley DBに格納

検索 隣接判定前の最適化 索引語の選択

- 隣接判定に使用する索引語を減らす
- 現在の最適化方法

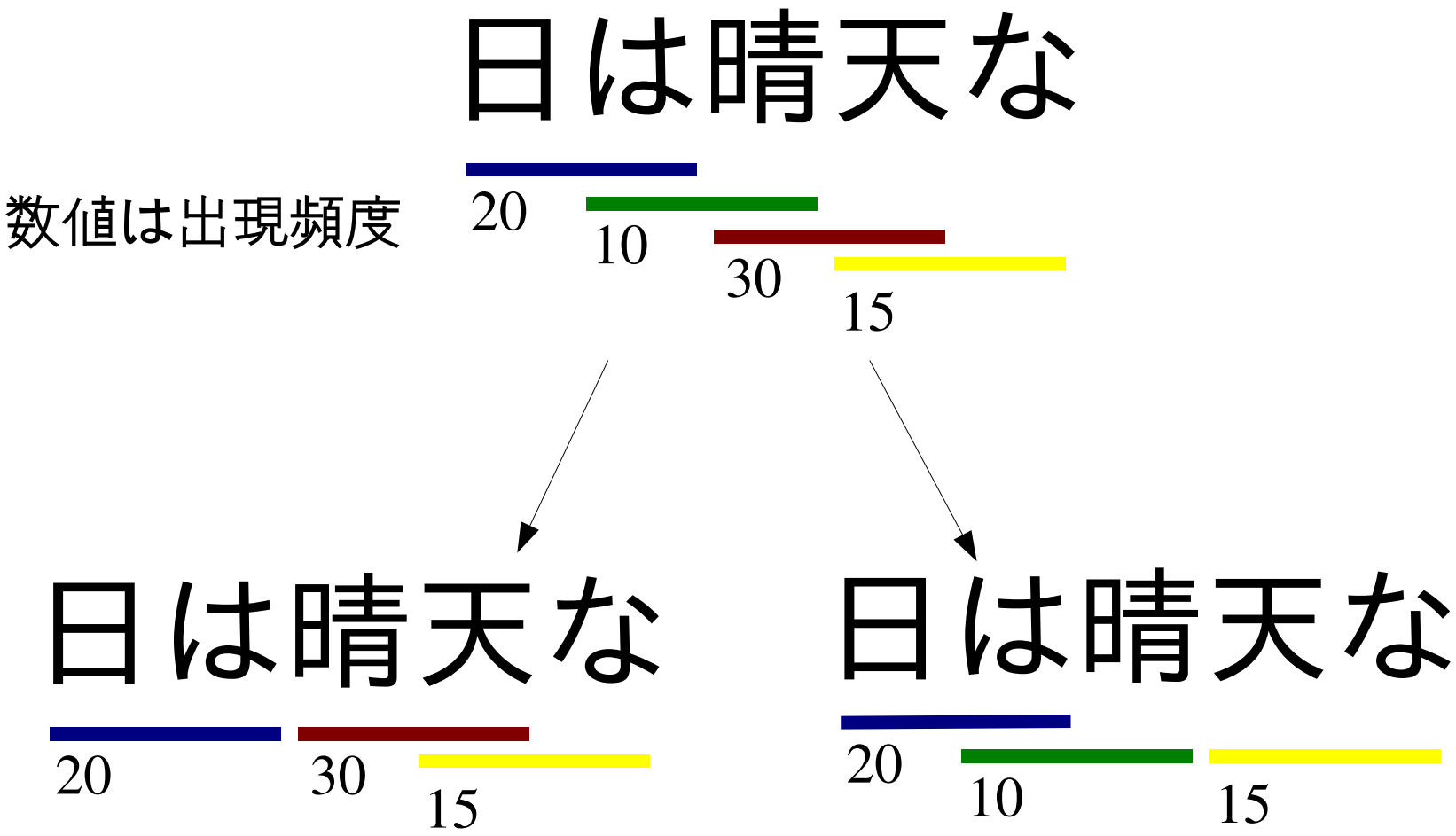
  1)基本的に先頭索引語から重ならない索引語を隣接判定に使用する

  2)出現文書数最少の索引語は重なっても使用する

  3)最後の索引語は重なっても使用する

検索 隣接判定前の最適化 索引語の選択

- 検索語 日は晴天な での隣接判定例



検索 隣接判定前の最適化 文書IDによる絞り込み

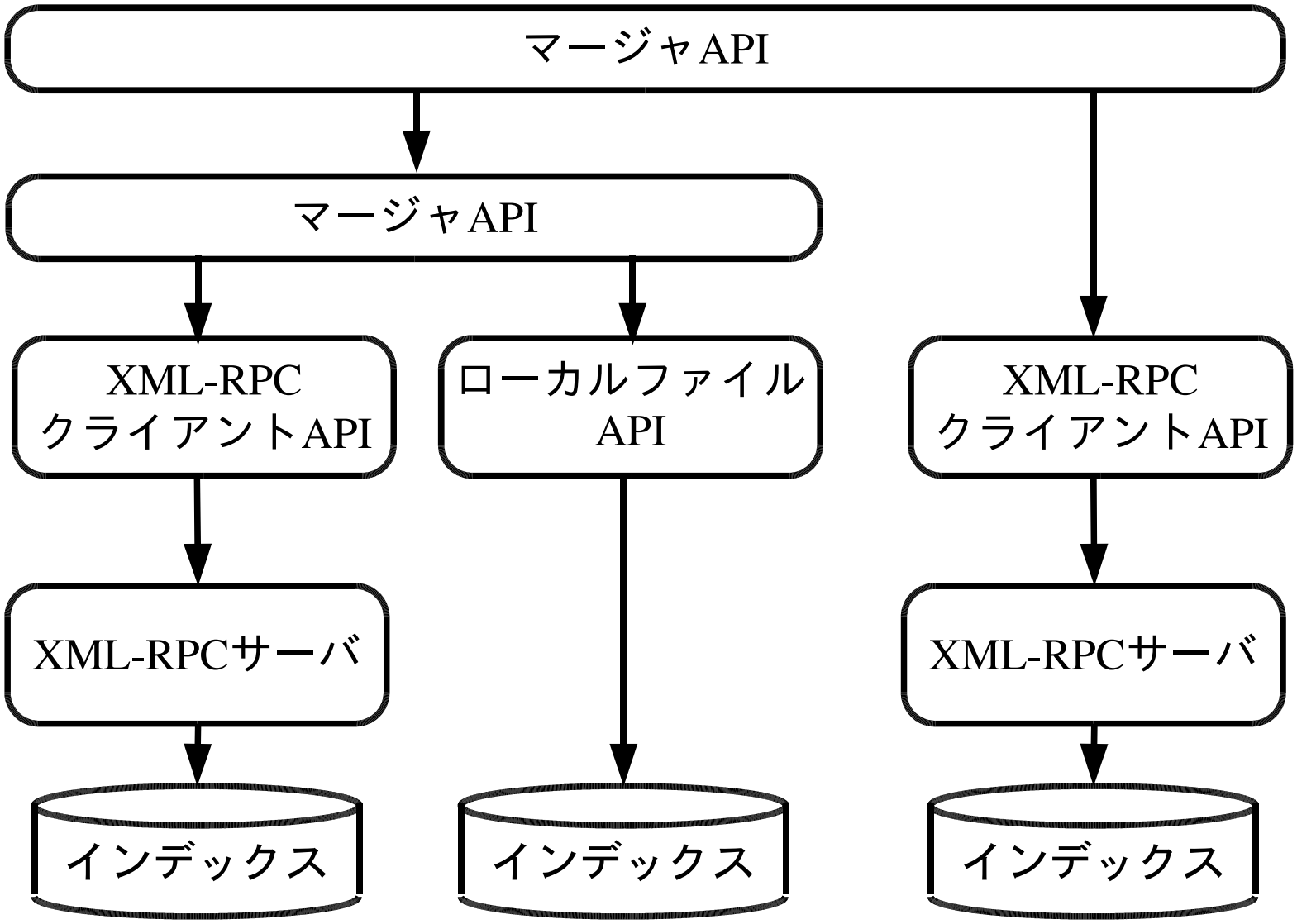
- 隣接判定前に, 全ての索引語に同じ文書IDが含まれていることを確認する.
  - 全ての索引語に同じ文書IDが含まれていないと隣接判定しない
- 検索語 本日は晴 での判定例
  - 3つの索引語 本日 日は は晴 に分割される
  - 文書1と4が選ばれる

	本文	本日	日は	は晴
文書1	本日は晴れです	○	○	○
文書2	本日東京は晴れ	○	×	○
文書3	本日は雨	○	○	×
文書4	本日は晴天なり	○	○	○

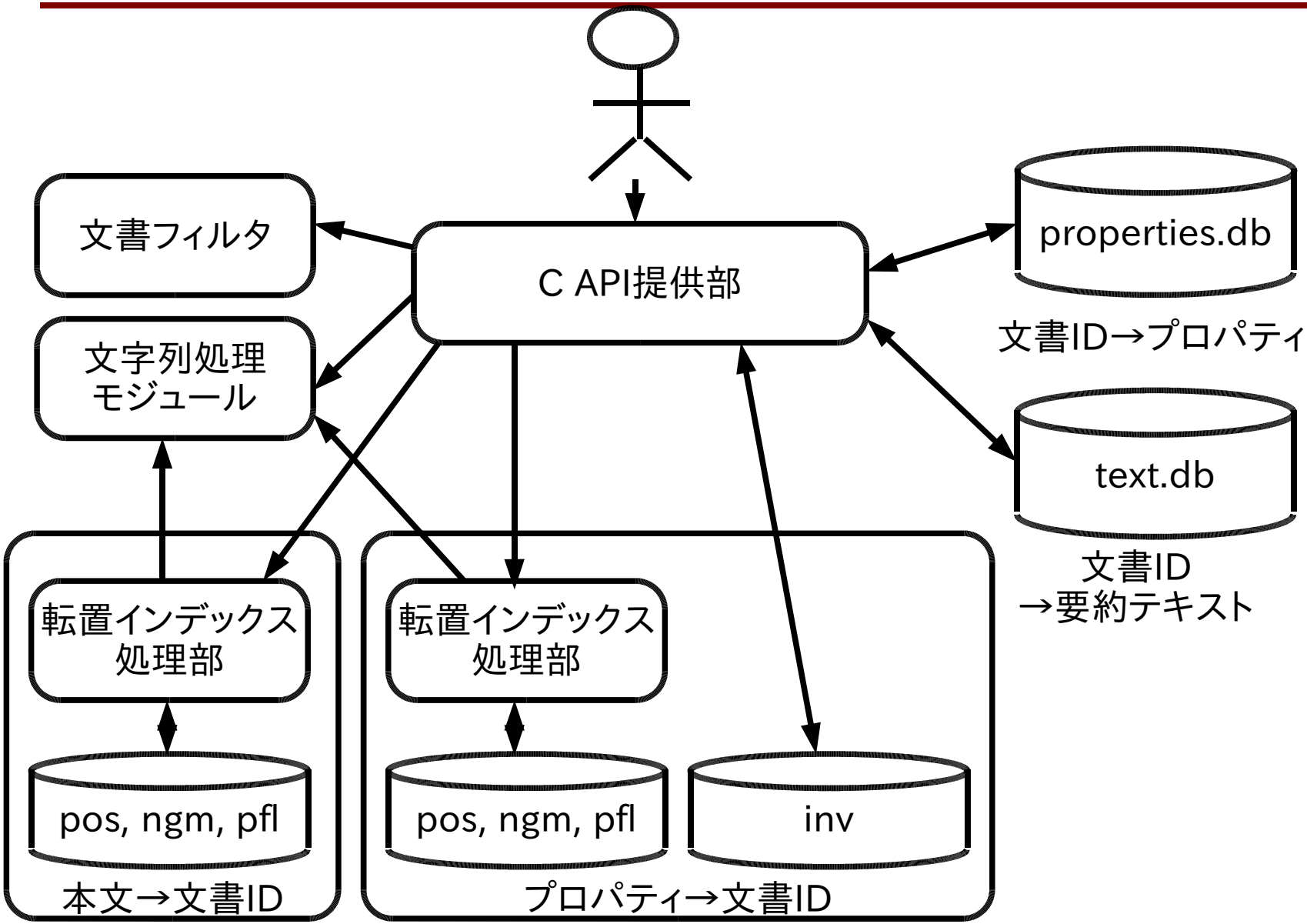
スコアリング

- 現状はTF-IDF法を検索語に適用.
  - (検索語の文書中の出現数 / 文字数) * (log10(全文書数 / 検索語の文書数) + 1) を 各検索語について行って加算し, intに収めるために1000000倍する.
  - おおよそ, 数10から数10万の範囲に収まる.
- 検索語の文書数を検索のたびに数えている
  - 分かち書きと違って検索語が索引作成時にわからないため

メタ検索 図


マージャAPI
マージャAPI
XML-RPC
クライアントAPI
ローカルファイル
API
XML-RPC
クライアントAPI
XML-RPCサーバ
XML-RPCサーバ
インデックス
インデックス
インデックス

本文の登録


文書フィルタ
C API提供部
properties.db
文書ID→プロパティ
文字列処理
モジュール
text.db
文書ID
→要約テキスト
転置インデックス
処理部
転置インデックス
処理部
pos, ngm, pfl
pos, ngm, pfl
inv
本文→文書ID
プロパティ→文書ID

本文の登録

- 登録操作

  1)バイト列を文字列処理モジュールを用いて索引語に分割し，索引語ごとの位置情報を得る．

  2)メモリ上のハッシュテーブルに格納する．

- sync操作
  - 1文書1箇所しか現われない索引語はtext.rngに文書IDと位置情報を格納する．
  - そうでない場合は，text.posに文書IDと位置情報を，text.ngmにtext.posの格納場所を格納する．
- インクリメンタルな登録も行える

Rastのインデックス

- 本文の全文検索用インデックス
  - text.ngm（text.pos中の格納場所情報ファイル）
  - text.pos（索引語位置情報ファイル）
  - text.rng（低頻度索引語位置情報ファイル）
  - text.pfl（未参照領域管理ファイル）
- プロパティ検索用インデックス
  - properties/プロパティ名.{inv,ngm,pos,rng,pfl}（プロパティ値→文書ID用インデックス群）
- その他
  - METADATA（インデックスの初期設定値），doc_info（文書IDの最大値，登録されている文書数），properties.db（文書ID→プロパティ値用インデックス），text.db（要約テキスト格納ファイル）

Rastのインデックス 全文検索用インデックス

- text.pos（索引語位置情報ファイル）
  - 各索引語の文書IDと出現位置を管理
    - 文書ID
    - 出現位置情報のバイト数
    - 出現位置情報（文字単位）
- text.ngm（text.pos中の格納場所情報ファイル）
  - 各索引語のtext.posでの格納位置を管理（BDB B木）
    - ブロックNo
    - ブロック数
    - バイト数
    - 文書数

Rastのインデックス 全文検索用インデックス

- text.rng（低頻度索引語位置情報ファイル）
  - 出現数の少ない索引語を管理（Berkeley DB B木）
    - 文書ID
    - 出現位置情報のバイト数
    - 出現位置情報（文字単位）

Rastのインデックス 全文検索用インデックス

- text.pfl（未参照領域管理ファイル）
  - 索引語位置情報ファイルの未参照領域を管理
    - ブロックNo
    - ブロック数

Rastのインデックス プロパティ検索用インデックス

- properties/プロパティ名.inv（大小比較検索及び完全一致検索用インデックス）
  - Berkeley DB B木
- properties/プロパティ名.{ngm,pos,rng,pfl}（部分一致検索用インデックス）
  - 本文の全文検索用インデックスと同じ構造をとる.

Rastのインデックス その他

- METADATA（インデックスの初期設定値）
  - ブロックサイズ
  - バイトオーダ
  - 本文も登録する/しない
  - プロパティの数とプロパティの定義
- doc_info（文書IDの最大値，登録されている文書数）
- properties.db（文書ID→プロパティ値用インデックス）
  - 検索結果でプロパティ値が要求された場合に使用する
- text.db（要約テキスト格納ファイル）

AND, OR, NOT 演算

- &, |, ! 演算子
  - AND, OR, NOT検索を行う事ができる
  - 演算子の優先順位は同じで左結合
  - 空白で区切られた場合より強い
    - 「A & B」は「A B」より結合強度が強い
  - ! 演算子は単項演算子ではなく, 二項演算子
- グルーピングするときは()を使う
- 特殊文字は""を使う

プロパティ

- 本文に関連付けて登録できる属性値
- 定義できるプロパティの種類
  - 文字列
  - 数値
  - 日付（現状では文字列と同じ）
- 検索時に使用できる演算子
  - 部分一致（:）
  - 大小比較（<, <=, >=, >）
  - 完全一致（=）
- min < property < max による検索も可能

要約表示

- インデックスが索引語の位置情報を持っている
- 正規化した本文中から最初の索引語の位置の前後指定された文字数を取り出す.

性能評価 評価観点

- 既存の全文検索システムと登録時間, 検索時間を比較する.
- 実用的な速度で登録と検索が行えることを確認する.

性能評価 測定内容

- NamazuとEstraierと比較する.
- 電子メイル(ruby-list, ruby-dev)の本文のみ63126件50611680文字(一件あたり平均801文字)の登録を行う
- あらかじめ検索システムが使う文字エンコーディングに変換しておく

性能評価 測定結果 登録

検索システム	Estraier	Namazu	Rast(N-gram utf8)
6万件登録時間（sec）	156.19	626.09	531.33
1件追加登録時間（sec）	8.7	23.81	7.94
データベースサイズ	139MB	116MB	324MB

性能評価 測定結果 検索

検索システム	Estraier	Namazu	Rast(N-gram utf8)
検索語	Ruby		
適合文書数	29547	N/A	50066
cold start (sec)	0.4318912	0.0347416	0.665318
hot start (sec)	0.076197	0.0010586	0.5553598
検索語	tk		
適合文書数	2232	612	4739
cold start (sec)	0.3576128	0.3562174	0.51231968
hot start (sec)	0.0183802	0.0346768	0.1419772
検索語	a		
適合文書数	N/A	3045	61918
cold start (sec)	0.2709114	0.6318622	15.1325726
hot start (sec)	0.009056	0.180159	10.3322752

課題 文字列処理モジュール

- いくつかの異なる処理が一つのモジュールに集中しすぎている
  - 文字エンコーディングを扱う処理
  - 索引語の切り出し単位
  - 文字列の正規化処理(要約用, 検索用)
- → 関数テーブルの継承による共通化
- 検索クエリでのASCII文字依存なパースを行っている
  - 非ASCII互換な文字エンコーディングが不可能
- → クエリに必要な特殊文字列の決定を文字列処理モジュールにもたせる

今後の予定 パフォーマンス関連

- 検索速度面 – さらなる隣接判定前の最適化
  - 出現文書数が最小になる索引語の組み合わせを選択する
  - AND演算やNOT演算時に各索引語の文書IDによる絞り込みを行う
- インデックスサイズ面
  - 指定バイト数以下の位置情報を低頻度索引語にする
- 登録面の改良
  - 指定した出現頻度以上の索引語については登録をやめる

今後の予定 機能追加

- 各種文書フィルタ
- トランザクション対応
- 類似文書検索
- 正規表現検索 ワイルドカード検索
- TF-IDF以外のスコアリング手法の実装

各種文書フィルタ

- dlopenされたタイミングでフレームワークに必要な情報を渡す
  - フィルタの名前
  - 処理可能なMIMEタイプ
  - 処理する関数ポインタ
- 各フィルタはストリームやメモリをラップするオブジェクト（apr_brigade）を受け取る
- テキストの抽出やプロパティの抽出を行う
  - 必要であればエンコーディング変換を行う
  - フレームワークがエンコーディング変換のためのヘルパ関数を提供する

各種文書フィルタ

- メタフィルタ
  - 指定したMIMEタイプのフィルタを呼び出しオブジェクト（apr_brigade）を渡す
  - MIMEタイプが指定されていなければlibmagicを用いて推測する
  - それでも不明な場合はエラーとする
- 登録フィルタ
  - 受け取ったテキストやプロパティを逐次データベースに登録する
- メイル, HTML, PDF, PostScript, OpenOffice.org, tar, zip...

ご静聴ありがとうございました

- Rastは全文検索の自由を提供します
  - 検索方式が選べる自由
  - 使用する文字エンコーディングが選べる自由
  - 全文検索機能をアプリケーションに組み込める自由
- http://www.netlab.jp/rast/
- 明日の検索システムBOFでデモします