Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

Cloud Search Engine as a Service

宮野 貴正[†], 上原 稔[†]

クラウドおよびビッグデータ時代ではサーチエンジンが重要である。特にクラウドに集約されたデータを効率よく検索するサーチエンジンが必要とされる。我々新鮮な情報検索に適した分散型サーチエンジン、協調サーチエンジン(Cooperative Search Engine, CSE)を開発した。また、クラウドに特化した協調サーチエンジンの改良版としてクラウドサーチエンジン(Cloud Search Engine, ClSe)を開発した。しかし、ClSe を利用するには、複数の部品を適切に結合して配備する必要がある。これは容易ではない。そこで、ClSe を手軽に利用するために all-in-one パッケージとして提供する。このようなサービス方式を我々は CSEaaS(Cloud Search Engine as a Service)と名付ける。本論文では CSEaaS 設計と実装について述べる。CSEaaS では、ClSe を容易に配備するため、簡易設定機能、柔軟な配備等を実現する。本論文では、典型的な 3 つの事例の構成を示す。

Cloud Search Engine as a Service

Takamasa Miyano[†], Minoru Uehara[†]

In cloud and big data era, search engine is important. Especially, data is collected to cloud now. So, search engine that search documents in cloud is required. We developed Cooperative Search Engine (CSE), which is a distributed search engine suited for fresh information retrieval. Furthermore, we have developed Cloud Search Engine (ClSe), which is based on CSE and extended for searching data stored into cloud. However, in order to use ClSe, it is necessary to provide ClSe as all-in-package and by deploying all components in suited way. This is not so easy. So, in this paper, we propose a service that provides ClSe as all-in-package in cloud. We call such a service CSEaaS (Cloud Search Engine as a Service). In this paper, we describe the design and implementation of CSEaaS. In CSEaaS, we realize easy setting and flexible deployment of ClSe. Furthermore, we show 3 typical use cases of ClSe.

[†]東洋大学 工学部 情報工学科
Dept. of Computer and Information Sciences, Toyo University.

1. はじめに

インターネットを利用する上でサーチエンジンの利用は不可欠である。サーチエンジンは Web の検索機能の欠如を補う形で発展した。初期のサーチエンジンはディレクトリ型であったが、今日では検索技術の進歩により全文検索型が主流となっている。サーチエンジンを用いて必要な情報を入手するスキルは現代情報社会におけるリテラシーと考えられている。

過去、サーチエンジンは活発に研究され、様々なシステムが開発された。しかし、今日まで残っているものは少ない。この理由としてサーチエンジンの決定打としての Google の存在があげられる。Google 以前は各サイトに独自のサーチエンジンを組み込む必要があった。しかし、Google 以後は Google のドメイン検索を利用するだけでよい。このため独自のサーチエンジンを開発する動機は失われた。

しかし、外部からアクセスできない内部情報の検索（すなわち Deep Web）にはサーチエンジンが不可欠である。また、近年では、データをクラウド上に配備することが多くなってきた。このようなデータも一種の内部情報である。今日では、ますますクラウドへデータが集約されつつあり、クラウド内の検索は重要な課題となってきている。しかし、クラウドのデータは Google 等外部のサーチエンジンに頼ることは困難である。それゆえ、クラウドを検索するサーチエンジンが求められる。

我々はクラウド上のデータを検索するために Cloud Search Engine(ClSe)を開発した。ClSe は IaaS 型クラウドにおける複数のインスタンス内の文書を効率よく検索するために協調サーチエンジン(Cooperative Search Engine, CSE)に基づく。CSE は新鮮な情報検索に適した分散型サーチエンジンである。Google 等の集中型サーチエンジンでは、ロボットあるいはクローラにより文書を収集する必要がある。しかし、分散型サーチエンジンでは、文書を収集する必要がなく、公開後ただちに検索できる。このために、分散型サーチエンジンでは、各サイトに小型のサーチエンジンを配備する必要がある。また、既存分散型サーチエンジンの課題は検索時の遅延であった。CSE ではこの問題を解決するため様々な高速化技法を取り入れ、集中型サーチエンジンとそん色ない性能を実現した。ClSe は Java による CSE の再実装であるが、加えて CSE で実現されていないサイトの耐故障性を実現している。集中型サーチエンジンでは、サイトが故障すると、検索はできるが、検索結果を閲覧できなくなる。分散型サーチエンジンである CSE では、検索も閲覧もできなくなる。しかし、ClSe では、インデックスの冗長化により検索を可能とした。文書自身も冗長化すれば閲覧も可能となる。

ClSe は IaaS 型クラウドの検索に適しているが、その配備には一定水準の知識を要する。Google の手軽さとは大きな違いがある。そこで、本論文では ClSe をサービスとして提供する方式 CSEaaS(Cloud Search Engine as a Service)について述べる。本方式では、最小限の ClSe を構成可能なイメージを提供し、IaaS インスタンスとして ClSe を

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

稼働する。インスタンスは設定により ClSe のいずれのコンポーネントにもなりえる。クラウドの規模に応じてスケールアウトさせることができる。

本文の構成は以下の通りである。2 節で関連研究としてクラウド、特に本研究で使用した教育用クラウドについて述べる。関連研究のうち特に重要な CSEt と ClSe については 3, 4 節で詳細に説明する。5 節では、CSEaaS の機能および実現について述べる。6 節では、その評価を行う。最後に結論を述べる。

2. 関連研究

2.1.1 クラウド

クラウドでは、スケールアウトと呼ばれる手法が使われている。スケールアウトとは安価なサーバーを多く利用し、分散処理させる事で性能を向上させる。また、仮想化と呼ばれる技術により、1 台の物理サーバーの中で複数台のサーバーを構築できる。そのため、サーバーのリソースを効率よく利用することが可能となった。

クラウドのサービスモデルは IaaS・PaaS・SaaS の３つに分類でき、配置モデルとしてプライベートクラウドとパブリッククラウドの２つに分類できる。

- IaaS

計算機やディスクなど、比較的ハードウェアに近い構成要素を仮想化して、利用者に提供します。利用者は、提供された仮想ハードウェア上にオペレーティングシステムやアプリケーションを自分でインストールして利用することになります。IaaS の代表的な例として Amazon の EC2 などが挙げられる。

- PaaS

クラウド利用者に対してアプリケーションを記述するためのプラットフォームを提供する。利用者は独自のアプリケーションを記述し、エンドユーザに対して提供します。特定のソフトウェアなどでなく、提供されるプラットフォームに自分でプログラムを構築し利用する。PaaS の代表的な例として Google の Google App Engine などが挙げられる。

- SaaS

ソフトウェアサービスをエンドユーザに直接提供する。Web ブラウザを経由してエンドユーザにアプリケーションを提供する。SaaS の代表的な例として Gmail や GoogleDocs などが挙げられる。

- パブリッククラウド

一般の不特定多数のユーザを対象とした配置モデルである。他に大きな括りでの産業体や組織体のユーザを対象としたクラウドの利用形態もこのモデルに含まれる。このモデルでは、多くの場合 IT 業界での大規模企業がクラウドプラットホームを所有し、一般ユーザにクラウドの利用基盤を公開・提供する。パブリッククラウドの代表例として、Google が提供する Google App Engine などが挙げられる。

- プライベートクラウド

企業や団体組織など単一の組織によって運用されるクラウドプラットホームである。この形態ではその組織内の要員または組織外の要員によって管理される場合の２つの形態があり、前者は自社運用型、後者は他社運用型と言われます。したがってプライベートクラウドの自社運用型はシステム構成がクラウドの形態をとる以外は、クラウド以前でのコンピュータ運用に最も近いものである。

2.1.2 教育用クラウド

教育用クラウドとはネットワーク上にあるサーバのサービスを端末から利用するクラウドコンピューティングの仕組みを、教育向けに特化したものである。既に複数の会社が提供しており、校務処理、成績処理、グループウエア、学校 Web サイト、デジタル教材などのクラウドサービスが提供されている。データを国内外のデータセンターに分散させて保管するなど、自然災害によるデータの消失を防いでいて安全性は高い。また、各種サービスを一元的に提供するものが多く、インタフェースが違ったり、費用がかさんだりして管理が複雑にならないというメリットがある。

しかし、教育用クラウドだが、普及するにはいくつか課題がある。第一に経費である。学校単位での導入でも意味はあるが、自治体単位で導入する方が効果は倍増する。グループウエアを利用することで他校との情報交換や共有が可能になるからである。ただし、自治体単位での導入は大きな経費が掛かかってしまう。第二に、個人情報の保護とセキュリティの問題がある。特に児童・生徒の成績や指導内容などの保護は完全でなければならない。データそのものは安全に管理されていても、簡単に不正アクセスされてしまえば同じである。第三の課題は、帳票類の統一である。成績処理ソフトを導入する場合、通知表をはじめとする帳票類の様式を全校で統一する必要がある。ただ、それぞれの学校にしてみれば、今まで使っていた様式が変わることは歓迎できないので、抵抗があるかもしれない。特に通知表は公簿ではなく学校で独自に決められるので統一は難しい。

3. 協調サーチエンジン

協調サーチエンジン(Cooperative Search Engine, CSE)とは，本研究室で研究されていた分散型アーキテクチャに基づいた分散型サーチエンジンである．特徴としては，複数の Web サーバ内にインストールされた局所的なサーチエンジンを，メタサーチエンジンが統合することにより，文書収集を不要とし，インデックスの更新間隔を短縮することができるサーチエンジンである．

CSE は図 1 に示されるように，大きく分けて 4 つのコンポーネントから構成されている．これらのコンポーネントを協調させる事により，更新間隔を短縮する事が可能

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

なサーチエンジンが構築できる．

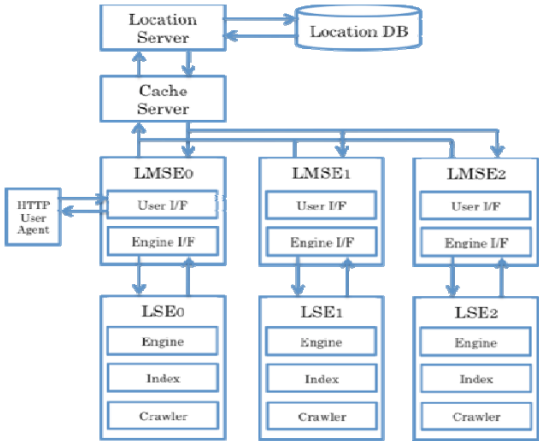

図 1. CSE の構成
Fig.1 The structure of CSE

- Location Server (LS)
  CSE では，局所的なサーチエンジンを作成し，そのインデックスのサマリ情報を一元管理する事により，検索時に検索したい局所的なサーチエンジンに検索要求をする事ができる．その情報を管理するのが LS である．また，LS は CSE 全体で 1 つだけ存在する．
- Cache Server (CS)
  CS は，検索結果などをキャッシュするサーバである．キャッシュをすることにより，「次の 10 件」のような継続検索を効率よくするために使用する．また，CS は後述の LMSE を並列に呼び出し，並列検索を行う．
- Local Meta Search Engine (LMSE)
  LMSE は，ユーザからの検索要求を受け付けて CS に転送する．また，後述の LSE を用いて局所的な検索を行う．
- Local Search Engine (LSE)
  LSE は局所的な文書収集，インデックスの作成，検索を行う．

CSE の更新時の動作は，以下のように行われる．

1. 各 LSE が対象サイトの文書を収集する．
2. 各 LMSE に対応する LSE は，収集した文書データからインデックスの作成を行う．インデックスの作成後，単語一覧とその単語を含む文書数と全文書数を自身の URL とともに LS に送信する．
3. LS は各 LMSE から送られた単語一覧とその文書数，全文書数をデータベースに保存する．

CSE の検索時の動作は，以下のように行われる．

1. ユーザからの検索要求を受け付けた LMSE は問い合わせ内容を CS に送る．
2. CS は LS に検索要求を満たす文書を持つ LMSE の集合を問い合わせる．
3. CS はその結果を用いて，各 LMSE に検索要求を送信する．
4. LMSE は LSE を用いて検索し，その結果を CS に返送する．
5. CS はこれらの検索結果をまとめて検索要求を受け付けた LMSE に返送する．
6. LMSE はユーザに検索結果を表示する．

4. クラウドサーチエンジン

我々は IaaS 型クラウド内の文書検索に適したサーチエンジンとしてクラウドサーチエンジン(Cloud Search Engine, ClSe)を提案した。

IaaS 型クラウド環境のように，大量のサーバを利用できる環境においては，CSE のようなメタサーチエンジンで複数のサーチエンジンを統合することにより，クラウドに適したサーチエンジンを構築することが可能と考えられる．だが，CSE をそのままクラウドで構築した場合の問題がある。

まず，IaaS 型クラウド環境でサーチエンジンを構築するためには，耐障害性を考える必要がある．CSE のコンポーネントの一部である LSE には，インデックスの冗長化が存在しない．それは，CSE が物理サーバにインデックスデータを保存する設計であるため，サーバが停止してもインデックスデータは残るためである．だが，IaaS 型クラウド環境では，インスタンスが停止してしまうとインスタンスに保存していたインデックスデータは消失してしまう．よって，クラウド環境でサーチエンジンを構築するにあたり，レプリケーション機能などで冗長化をする必要がある．

また，IaaS 型クラウド環境では，大量のサーバを用いているため，スケールアウト可能な設計がとても重要である．だが，LS は CSE 全体で 1 つだけ存在するために，負荷が集中してしまう．よって，スケールアウトが可能な NoSQL の分散データベースを用いる事により，負荷を軽減することができると考えられる．また，LSE は Consistent Hashing でインデックスの作成を分散させる．そのため，新たにサーバが追加されても一部のデータを再配置するだけで通信のトラフィックを最小限に抑える事ができると考える．

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

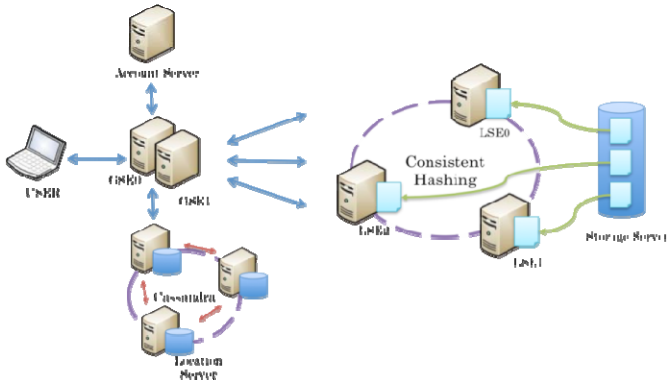

図 2. ClSe の構造
Fig.2 The structure of ClSe

ClSe の構成は，図 2 のように 5 つのコンポーネントから構成される．以下がコンポーネントの説明である．

- Local Search Engine (LSE)
  LSE は，実際の検索を担当する局所的なサーチエンジンである．また LSE は複数台でシステムを構築し，インデックスをレプリケーションする事で一部の LSE が停止してもシステムは可動し続けられるようにする．ClSe では、LSE として Apache Solr を採用している。
- Global Search Engine (GSE)
  ユーザからの検索要求を受け付けるサーバである．また LSE に対して分散検索を依頼し，その結果をユーザへ返す．ユーザに結果を返す際に，ランキングの修正を行う．ClSe では、GSE として Apache Solr を採用している。
- Account Server (AS)
  アカウント情報を管理するサーバであり，アカウント毎にアクセスできる文書の範囲を管理するサーバである．LSE のインデックスデータは，全てのアカウントをまとめてインデックスが作成される．そのため，アカウント毎に検索できる範囲の文書データを決める必要がある．
- Location Server (LS)
  LS の検索時には，ユーザからの検索キーワードを基に，実際にインデックスデータを持っている LSE のサーバを決める．また更新時には，各 LSE のインデックスの単語一覧とその単語を含む文書数また全文書数をデータベースに保存する．ClSe では LS として Apache Cassandra を採用している。
- Storage Server(SS)

ストレージサーバは，検索対象となるオリジナルの文書データを保存するサーバである．クラウドではインスタンスが停止してしまうと，それまで保存していたデータが消失してしまう．よって，消失しては困るオリジナルデータはストレージサーバを用いる事にする．また本システムでは，NFS のような分散ファイルシステムで構築する事を前提としている．

5. CSEaaS

ClSe 自体は優れたシステムであるが、それを利用することは決して容易なことではない。複数の部品で構成されるために個別の設定を理解する必要がある。また、運用が長期化すると多くのデータが蓄積され、資源が飽和する。飽和しない十分な資源を割り当てる必要がある。また、検索結果を参照する際、検索された文書に Web からアクセスする必要がある。

CSEaaS の要件をまとめると以下のようになる。

- Policy based configuration
  管理ポリシーを一元的に適用可能とする。XML 形式のポリシーファイルから各種設定ファイルを自動生成する。設定ファイルには生成の過程を通じてポリシーが強制的に適用される。また、同時に複雑な設定を理解しなくても、容易に設定が可能となる。
- Provisioning
  あらかじめ十分な資源を論理的に割り当てる。Provisioning は VMM レベルで実現されるため、本提案のサービスとは実装レイヤーが異なる。そこで、本論文では、Provisioning は除外する。
- Web based access
  検索した文書を Web から参照できる。元の文書は必ずしも Web で公開されたものとは限らないので、Web にアクセスのための代替経路を設ける必要がある。

CSEaaS は all-in-one のイメージとして提供される。1 つのインスタンスとして実行することもできるし、複数のインスタンスで役割分担して実行することもできる。

ここでは、典型的な 3 種理のケースについて CSEaaS の構成を示す。

第 1 のケースは、もっとも単純な構成である。この構成では、CSEaaS のインスタンス上に文書を格納する。組織内で文書収集する場合にも同じ構成となる。ただし、分散型検索では文書収集は必須の機能ではない。それゆえ、CSEaaS では文書収集の機能はオプションとする。

図 3 に第 1 のケースの構成図を示す。この図では、すべての部品が一つのインスタンスにインストールされている。

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

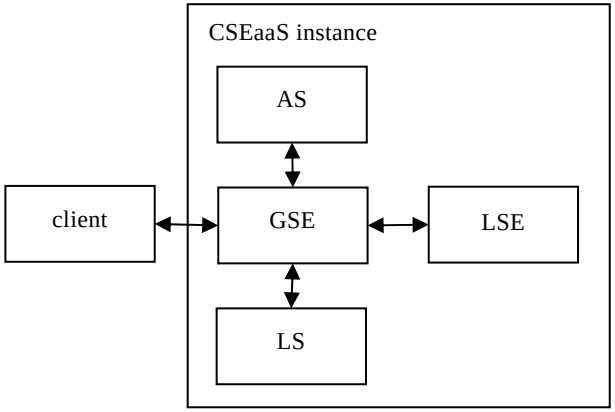

図 3. 第 1 のケース
Fig.3 First Case

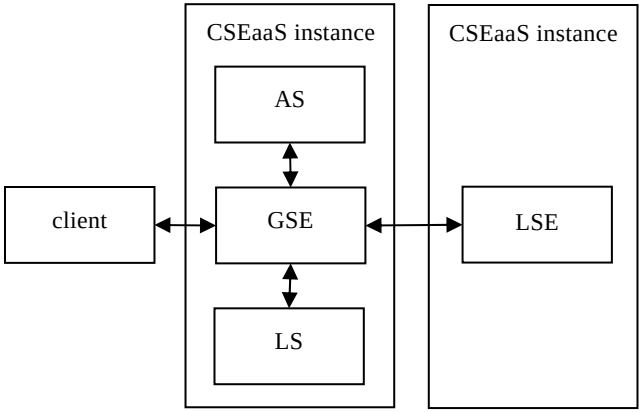

図 4. 第 2 のケース
Fig.4 Seconds Case

第 2 のケースは、文書とサーチエンジンを分割する構成である。この構成では、LSE とその他のコンポーネントが分割される。LSE は文書サーバに配備される。文書サーバが複数存在する場合も基本的な構成は変わらない。

図 4 に第 2 のケースの構成図を示す。この図では、LSE が別のインスタンスに分割されている。LSE があれば、そのインスタンスの文書を検索することができる。LSE は各インスタンスに 1 つあればよい。また、異なる複数のインスタンスにそれぞれ存在してもよい。LSE が稼働するインスタンスは CSEaaS のインスタンスを同一であるが、その他の分品が無効化されている。

ポリシーはインスタンス単位で適用される。そのため、複数のインスタンスを用いる構成では、個別にポリシーファイルを用意する。ポリシーファイルは XML ファイルであるため、全体のポリシーから自動的に生成することも可能である。

第 3 のケースは、耐故障性を考慮した構成である。耐故障性の段階に応じて細分化される。3-1 では、GSE が冗長化される。3-2 では、LS が冗長化される。3-3 では、LSE も冗長化される。

図 5 に 3-2 のケースの構成図を示す。この図では、LS が別のインスタンスに分割されている。LS が稼働するインスタンスは CSEaaS のインスタンスを同一であるが、その他の分品が無効化されている。同様に 3-1, 3-3 のケースも実現できる。

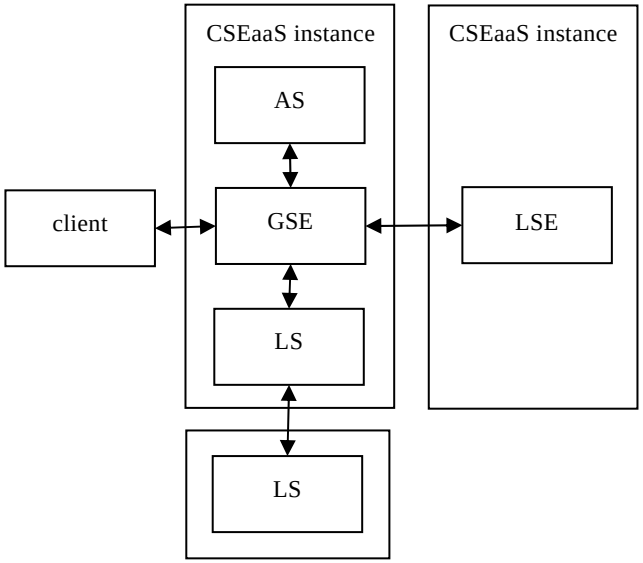

図 5. 3-2 のケース
Fig.5 Case of 3-2

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

6. 評価

CSE についての評価は文献で述べられておる。ClSe についての評価は文献で述べられている。ここで、CSEaaS について評価する。

ポリシー設定が可能であることを示す。

Java による XML 形式のポリシーファイルから各種設定ファイルを自動生成して、設定ファイルには生成の過程を通じてポリシーが強制的に適用されるには DOM という API を使用すれば実行可能である。

DOM とは、プログラミング言語に依存しない API であり、XML 文書に対応するツリー構造を静的にメモリ上に保持してランダムアクセス可能な API である。構造に対する変更や、プロセッサによる解析結果にランダムアクセスする必要があれば、DOM によってすっきりとしたアプリケーション・コードが記述できる。

```
//ドキュメントの作成
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
DocumentBuilder db = dbf.newDocumentBuilder();
Document document = db.newDocument();
Element book = document.createElement("gse-config");
document.appendChild(book);
<ルート要素名>
Element chapter = document.createElement("location");
Element add = document.createElement("host");
add.appendChild(document.createTextNode("192.168.168.234" ));
<ノードの作成>
chapter.appendChild(add);
Element port = document.createElement("port");
port.appendChild(document.createTextNode("9160" ));
chapter.appendChild(port);
book.appendChild(chapter);
```

図 6 DOM
Fig 6. DOM

第 1 から第 3 のケースが構成可能であることを示す。

第 1 のケースと第 2 のケースは構成可能である。検索するにあたってドキュメントファイルを登録した。はその検索結果である。

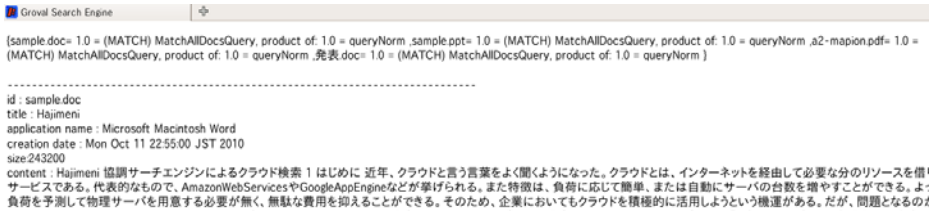

図 7 検索結果
Fig 7. search results

検索結果ではファイル名・ファイルの種類・サイズ(KB)・本文を表示させている。第 3 のケースはまだ構成途中である。

7. まとめ

本論文では、本件研究では協調サーチエンジンの仕組みを用い CSEaaS の作成を行った。第 1 のケースと第 2 のケースは構成可能で検索が可能であったが、第 3 のケースはローカルの LS に通信ができなかった。また、Cassandra や Solr など複数のコンポーネントからなっているので、その設定を統一的に行うために XML ファイルで各コンポーネントにアクセスアドレスの設定を自動生成することができるようになった。

今後の課題として、第 3 のケースでのクラウド内によける LS とローカル環境における LS との通信の実装する必要である。XML ファイルにより各コンポーネントの設定を同時に複雑な設定を理解しなくても、容易に設定が可能とすることができるようにする。

参考文献

1) Shinichiro Kibe, Minoru Uehara: "Proposal for a Cloud-based Educational Environment", In Proc. of 3rd International Workshop on Information Technology for Innovative Services(ITIS2011) in conjunction with the 14th International Conference on Network-Based Information Systems(NBiS2011), pp.523-528, (2011.9.7-9,Tirana,Albania)
2) Shinchiro Kibe, Minoru Uehara, Motoi Yamagiwa: “Evaluation of Bottlenecks in an Educational Cloud Environment“, In Proc. of the 13th International Symposium on Multimedia Network Systems and Applications (MNSA2011) in conjunction with the 3rd IEEE International Conference on Intelligent Networking and Collaborative Systems (INCoS2011), pp.520-525, (2011.11.30-12.2, Fukuoka, Japan)
3) Yuuta Ichikawa, Minoru Uehara: "Distributed Search Engine for an IaaS based Cloud", In Proc. of the 6th International Conference on Broadband, Wireless Computing, Communication and Applications(BWCCA2011), pp.34-39, (2011.10.26-28, Barcelona, Spain)
4) Yuuta Ichikawa, Minoru Uehara: "Cloud Search Engine for IaaS", LNCS TCCI (TBA)

Vol.2012-DPS-150 No.45
Vol.2012-CSEC-56 No.45
2012/3/1

5) Nobuyoshi SATO, Takashi YAMAMOTO, Yoshihiro NISHIDA, Minoru UEHARA, Hideki MORI "Information Retrieval Method for Frequently Updated Information System", Subhash Bhalla(Ed.), "Databases in Networked Information Systems", International Workshop DNIS 2000, LNCS1966, Springer-Verlag, pp.188-199, (2000.12)
6) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, "Fresh Information Retrieval in Cooperative Search Engine," In Proceedings of the ACIS 2nd International Conference on Software Engineering, Artificial Intelligence, Networking & Parallel/Distributed Computing(SNPD'01), pp.104-111, (2001.8.20)
7) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, "Fresh Information Retrieval using Cooperative Meta Search Engines," In Proceedings of the 16th International Conference on Information Networking (ICOIN-16), Vol.2, 7A-2, pp.1-7, (2002.1.31)
8) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, "Persistent Cache in Cooperative Search Engine," In Proc. of The 4th International Workshop on Multimedia Network Systems and Applications(MNSA'2002), in conjunction with The 22nd International Conference on Distributed Computing Systems(ICDCS'22), pp.182-187, (2002.7.3)
9) Nobuyoshi Sato, Minoru Udagawa, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, "Reliable Distributed Search Engine based on Multiple Meta Servers," In proceedings of IEEE 2002 International Symposium Cyber Worlds: Theory and Practices(CW2002), pp.79-84, (2002.11.6-8)
10) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, "A Case Study on Freshness based Scoring for Fresh Information Retrieval", In Proceedings of IEEE International Symposium on Communications and Information Technologies 2004(ISCIT2004), pp.210-215, (2004.10.27)
11) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, "Persistent Cache in a Distributed Search Engine" In Proceedings of 5th International Workshop on Network-Based Information System(NBiS), pp.54-58, (2002.9.2-6)
12) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, "The Evaluations of FTF-IDF Scoring for Fresh Information Retrieval," In Proceedings of IEEE 19th International Conference on Advanced Information Networking and Applications(AINA2005), pp.635-640, (2005.3.29)
13) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai "Evaluations of Freshness Considering Scoring on Fresh Information Retrieval", Journal of Interconnection Networks Vol.6, No.3, World Scientific Publishing Company, pp.265-282,(2005.9)
14) Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori: "Redundant architecture in Cooperative Search Engine", Int. J. Applied Systemic Studies, Vol.3, No.1, pp.73-88, 2010