議事録アノテーションによる議論の支援と再利用に関する研究

清水　敏之

名古屋大学 工学部 電気電子・情報工学科

2003年2月

目 次

概要

複数の人間が集まる会議や討論会などの場において、スムーズに議論が進むこと、さらに、その議事内容を検索し再利用できるということは非常に有益である。しかし、オンラインの会議、つまり、電子メールや電子掲示板などによる会議とは異なり、オフラインの会議、つまり、物理的な場所に集まって対面的に行う会議においては議論を再利用可能な形式にするのは一般に困難である。

そこで、本研究では、オフラインの会議における議事録の生成と再利用を支援するシステムを試作し、実験を行った。

本研究で提案するシステムは、構造化された議事録データを半自動的に生成し、リアルタイムに可視化して表示することができる。作成された議事録には、他の議事録に対する参照関係や発言内容の言語構造に関するアノテーションを付与することができる。また、その議事録を XML データベースおよびリレーショナルデータベースに登録し、キーワード等で検索した後に、アノテーションに基づいて計算された発言の重要度を用いた要約等の作成を可能にした。

第1章　はじめに

1.1　議事録の有効性

関係者が集まり、討論・相談や決議をする会議という場において、スムーズに議論が進むということ、さらに、過去の議事内容を検索などの手段により参照して、再利用できるということは非常に有益である。「議事録」という形で議論内容を保存しておかないと、過去の議論は容易に忘れ去られ、何度も同じ議論を繰り返す恐れがある。さらに、過去の議論内容を参照したい場合も、あやふやな記憶から間違った内容を新たな議論の場に出してしまうかもしれない。これは、現在進行中の会議においても同様のことが言える。議論の途中で、現在までの議論の内容を確認できるということも同様に有益である。今までの議論を一度振り返ることで、人間の創造性を刺激することができる。しかしながら、オンラインの掲示板などによる会議とは異なり、オフラインの会議においては議論を再利用可能な形にするのは一般に困難である。

1.2　本研究の目的

本研究の目的はオフラインの会議に焦点をしぼった、議事録の生成と再利用にある。議事録の生成を支援する仕組み、生成する際に議事録につけられた発言に関する付加情報アノテーションを利用した議事内容の再利用の仕組みを考察し、そのためのシステムを提案する。

1.3　本研究で提案するシステムの概要

本研究で提案するシステムは、アノテーションをつけることにより、構造化された議事録データを半自動的に生成し、XML データベースとリレーショナルデータベースに登録する。議事録の入力としては Web ブラウザのフォーム上でのテキスト入力とし、スライドなどの情報も含まれる。また、発言のグラフ表示・編集モードを設け、議事録における発言構造をグラフィカルに見ることができるとともに、議事録データに対するアノテーション作成を補助することもできる。データベースに登録された過去の議事録データを容易に検索し、その内容やグラフ構造を表示することが可能である。さらに、アノテーションを付けられた議事録データは要約などの再利用が可能である。以上のような機能により議論を支援するシステムを試作した。

1.4　本論文の構成

以下第２章では、本研究で提案するシステムにおいて、どのように議事録を生成し、アノテーションを付け、構造化するかについて述べる。第３章では、生成された議事録データを再利用する仕組みについて示す。第４章では、いくつかの関連研究を紹介し、最後に第５章で、本研究のまとめと今後の課題について述べる。

第2章　議事録の生成とアノテーション

会議の場で議事録を生成する際、どのように記録をとっていくのがよいだろうか？

１つは、紙に書き留めてゆくという手段である。紙というメディアは我々にとって慣れ親しんだものであり、速記などの手段で発言を書き留めることで、かなり正確な記録が期待できる。しかし、紙というメディアは再利用性が低く、検索などが可能な状態にするには、一度電子化する必要がある。電子化するには、かなりの人手と労力が必要であると思われ、この「紙に書き留めていく」という手段は、後々の再利用性のことを考えると適切でない。

それならば、キーボードなどによりパソコンに直接入力していく、という手段が考えられる。電子的に記録をとることで、再利用性を高めることができる。基本的にはこの手法でよいと考えた。

議事録を入力していくにあたっては、日付や参加者情報など、議論内容の他にもいろいろな情報を記録しておくべきであり、後々の再利用性の高さが期待できることから、本研究では議事録を XML (Extensible Markup Language) [1] 形式で記述することにした。

この XML 形式で書かれた議事録（以下議事録 XML）を生成するにあたり、まず考えられる手段が、タグなども含んでテキストエディタに直接入力していく、という手段である。この手段は特別なものは何も必要ない、という利点がある。しかし、タグ名の誤り、タグの不整合、構造の不一致、といった問題が発生する。これらの問題は人が直接入力するという性質上、避けられない。

次に考えた手段は、議事録生成用の GUI を作成し、そこから入力するというものである。これの利点は、テキストエディタで直接入力していくときに問題となった、タグ名の誤り、タグの不整合、構造の不一致、といった問題が発生しないことである。さらに、入力の効率の向上も期待できる。これの問題点は、議事録を入力する人がその入力用 GUI のプログラムをいちいちインストールして実行しなければならないことである。

以上のことを踏まえて、本研究ではブラウザのフォーム上からのテキスト入力により、議事録 XML を生成する、という手段をとった。この手段では、ユーザの側は何も特別なプログラムをインストールする必要はなく、XML 形式でデータを記録する際に起こる、タグ名の誤り、タグの不整合、構造の不一致、といった問題も発生しない。

議事録を有効に再利用するには、議事録を作成する際にどれだけ良いアノテーションをつけることができるかにかかっている。本章では本研究で試作したシステム（以下本システム）において、議事録が生成され、それにアノテーションが付与されるまでの流れを示す。

2.1　スライド情報の取得

本システムでは、スライド情報に特に注目した。

近年の会議においては資料として Microsoft PowerPoint で作成されたスライドを用意する場合がほとんどであろう。このことを利用し、議事録生成の手助けとする。

本システムでは PowerPoint などの COM (Component Object Model) を Java で操作するために JACOB (A Java-COM Bridge) [2] を利用した。COM とは再利用可能で様々な言語で利用できる Microsoft が策定したオブジェクト間の通信規約のことであり、Windows 上で複数のアプリケーションやソフト部品を連携させることができるものである。JACOB とは Java から Windows の COM を利用することのできる、Java-COM ブリッジであり、Excel や Word などの既存の COM の機能を Java から利用することができる、というものである。

本システムにおいては PowerPoint によって製作されたスライドは必須である。PowerPoint のスライド以外の資料を用いて議論を行う際も議題を記入したスライドを用意して議論を行う。その場合は、多少の冗長さを伴うが、そのスライドから議題の情報が自動的に議事録に追加される。

Microsoft PowerPoint であらかじめ発表者がその会議のために作成しておいたスライドを指定すると、議事録サーバ側で JACOB が PowerPoint を呼び出し、その PowerPoint のスライドの一枚一枚を GIF 形式のイメージに変換する。

そして、一枚目のスライドからタイトル情報を取り出し、それを議題として議事録に追加する。さらに、スライド一枚一枚から、そのスライドのタイトルとそのスライドに含まれるテキストを取得し、議事録に追加する。

GIF 形式のイメージに変換されたスライドは議事入力ページに表示され、議事入力の際にスライドと発言を関連付けて入力することを可能にする。このことにより議事録の構造化を助ける。

2.2　議事録の入力

本システムにおいて、議事録は主にブラウザのフォーム上からのテキスト入力によって生成される。その際、入力フォームは細かく分かれており、議論の内容だけでなく、発表者、日付、議題、参加者などの情報も入力する。

議事録を入力していくにあたり、書記の役目を行う人間が必要である。その書記が議事録をとっていく。

議論が始まると、まず議題提供者の ID と議論が行われる日付を入力する。この 2 つの情報は、その後、その議論を識別するのに使用される。日付情報はデフォルトで自動で今日の日付が入り、ほとんどの場合、そのままでよい。

そして、「詳細情報」ボタンを押して詳細情報入力ページ (図 2.1) を開く。

この詳細情報入力ページでは、その議論の基本情報とも言える情報を入力する。具体的には、書記の ID、議題提供者の ID、日付、議題、使用するスライド、議題提供者の詳細情報（名前、所属、メールアドレス、ホームページ）、参加者情報を入力する。

このうち、議題提供者の ID と日付は、先のページで入力したものが入り、使用するスライドと議題は、先のページで指定したスライドのファイル名とそのファイルから自動で抜

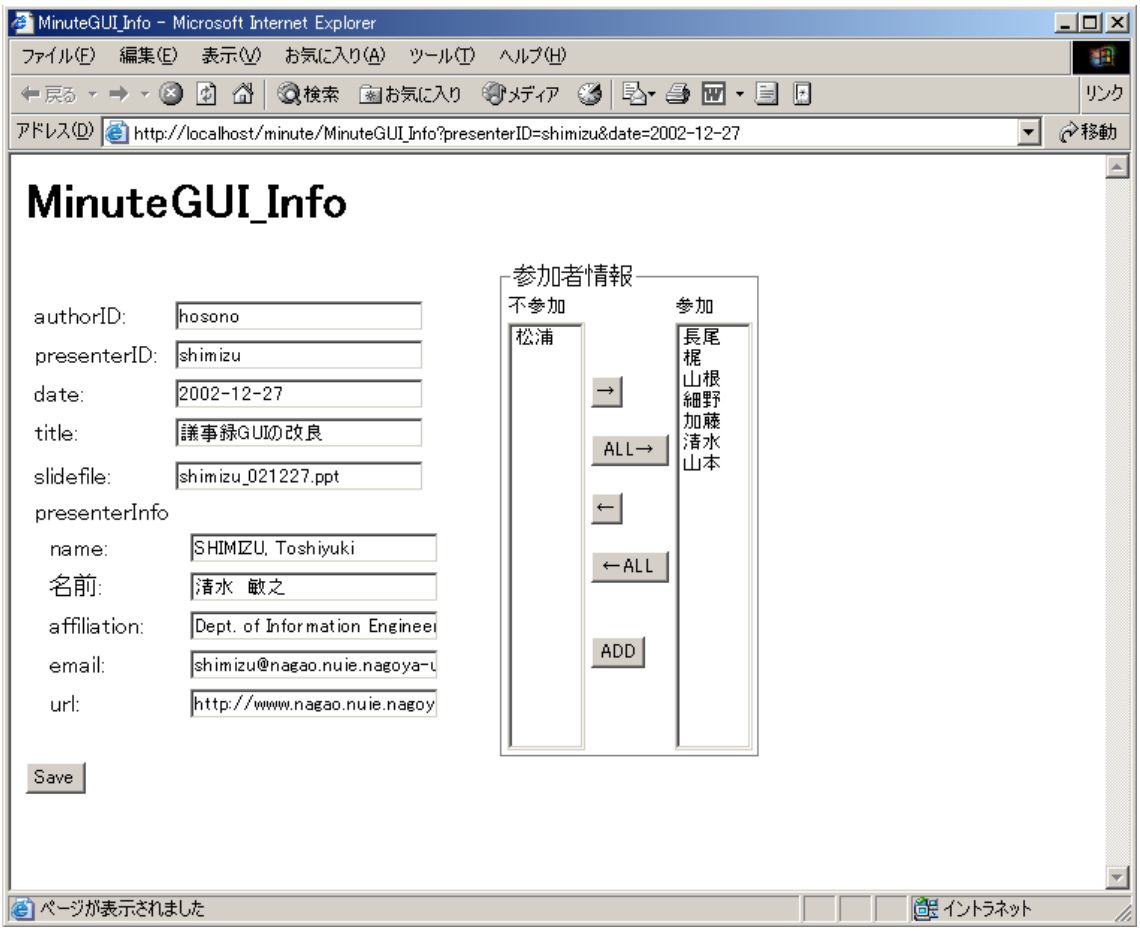

図 2.1: 議事詳細情報入力ページ

き出されたタイトルが入る。議題提供者の詳細情報は登録してある人のものだったら、設定ファイルから情報を読み込み、自動で入る。

よって、この詳細情報入力ページで、実際に手動で入力する情報はほとんどの場合、書記の ID と参加者情報だけである。

参加者情報を入力するエリアは、JavaScript による入力補助の仕組みが埋め込まれている。初期状態では設定ファイルに登録された人の名前が全員分「不参加」のフィールドに表示されている。その状態から、参加している人を選択して「→」ボタンを押すことによりその人を「参加」フィールドに移動させ、参加状態にすることができる。登録された人が全員参加している場合は「ALL →」を押すことにより全員を「参加」フィールドに移動させ、参加状態にすることができる。また、逆に移動させる効果を持つ「←」と「← ALL」ボタンを実装し、これらのボタンを組み合わせて使用することで、比較的楽に参加者情報を操作することが可能である。登録されていない人が参加している場合は、「ADD」ボタンを押すことによりでてくる入力ダイアログに名前を入力することで、参加者を追加することができる。

詳細情報ページへの入力が終了したら、「Save」ボタンを押して情報を保存する。

議論が始まると書記が発言を書いていくことになる。本システムでは、発言はスライドのどれか、またはスライド間のどこか、に関連付けて入力する。発言入力フォームがスライドの横とスライド間に用意されており、適切な位置に入力していく。

さらに、発言の情報は「質問」「回答」「コメント」の 3 種類に分類して入力する。本システムにおいては発言の内容をテキストで入力するので、煩雑になってしまわないよう発言の分類を大きく 3 種類にとどめた。

議論は「質問」「回答」「コメント」の 3 種類で 1 セットであると考え、一つの話題からもう一つの話題に変わると、新たな入力フォームを出せるような仕組みを実現した。適切な位置の「ADD」ボタンを押すことでこれを実現できる。

また、誤って入力フォームを出してしまったり、間違った情報を入力してしまって、入力を取り消したい場合は、「DEL」ボタンを押すことで入力フィールドを消すことができる。

議論の内容については以上のような手順で入力していく。

さらに、参照 URL と参照ファイルを入力するスペースも用意した。議論の途中で参照した URL やファイルのパスと、それに対する説明をこのスペースに入力する。これらの情報は複数入力される可能性があるので、発言の入力フォームと同様に「ADD」ボタンと「DEL」ボタンを用意して、複数の入力に対応した。

最後に、行われた議論に対する評価を入力する。これは 5 段階評価で入力して、後々の反省材料に使用される。

この他に、書記が使用できるフリースペースを用意した。このスペースは書記が自由に使用してよい。このスペースに書記はメモを取ることができる。

図 2.2 に議事録の入力画面を示す。

2.3 議事録へのアノテーション

アノテーションとはコンテンツに対して付加情報をつけることであり、本システムでは、入力された発言に対して発言同士の関連付けをするというアノテーションを行った。アノ

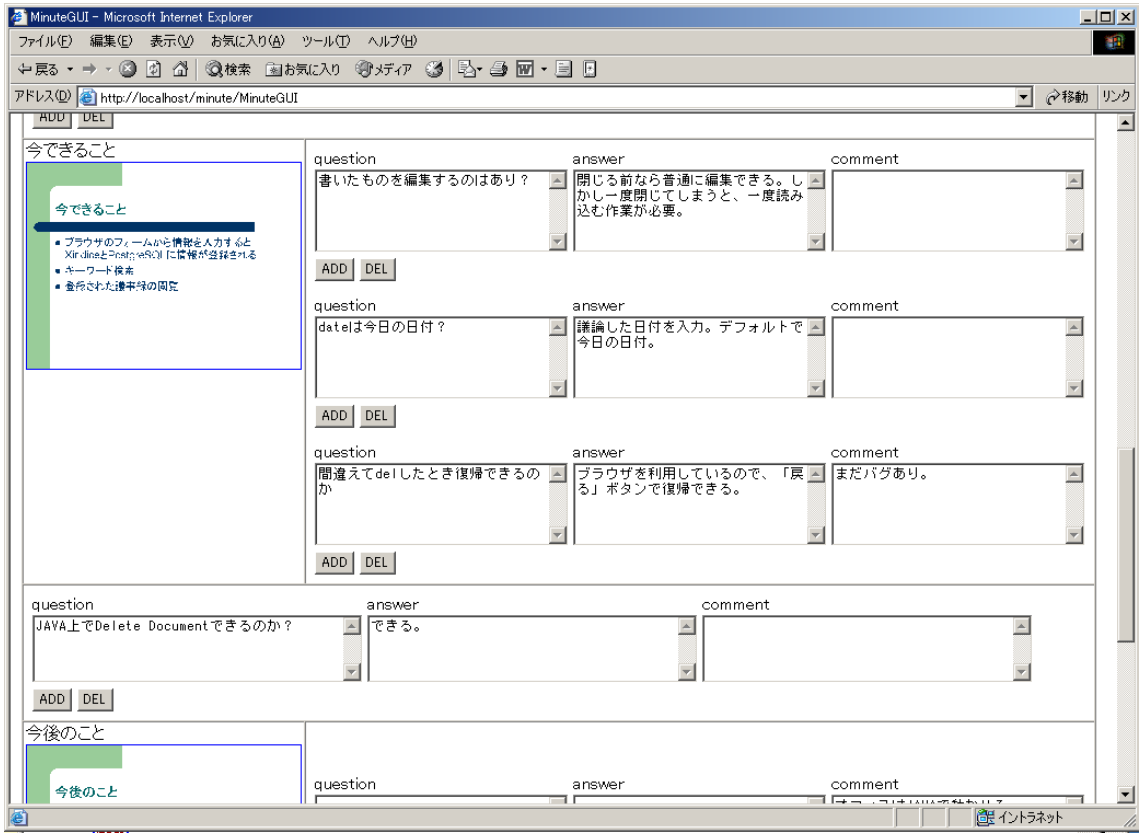

図 2.2: 議事録の入力作業

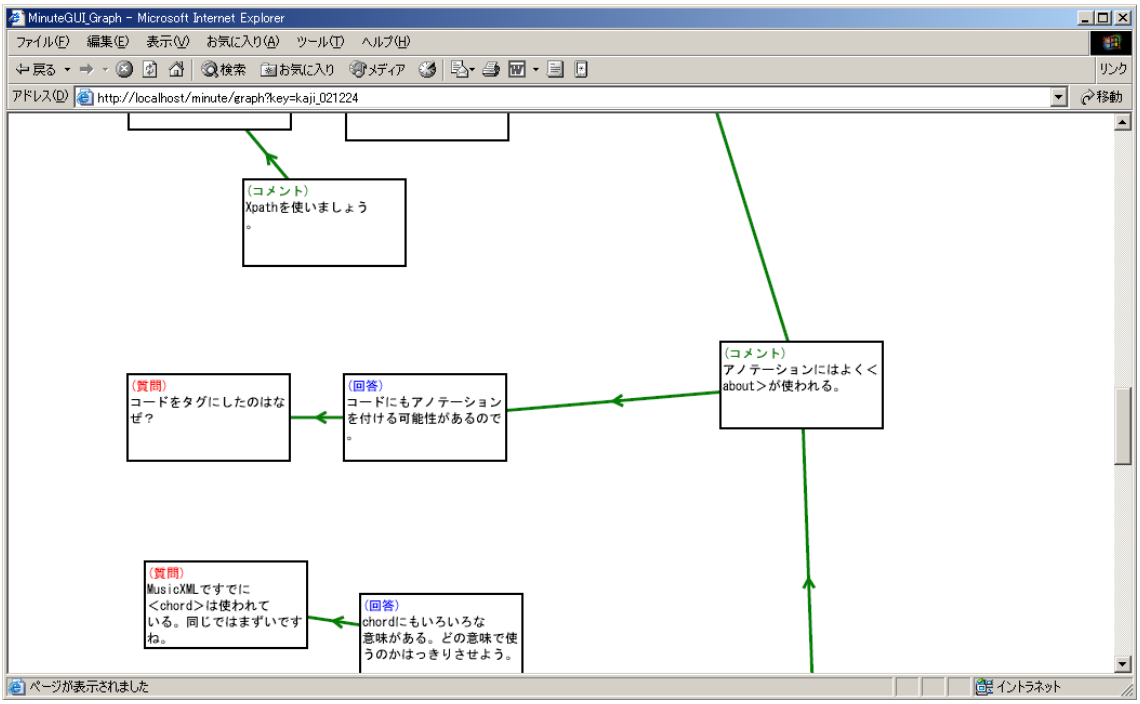

図 2.3: 議事録のグラフ表示

テーションにはある程度人間が介入する必要があり、少なからず労力が必要であるが、アノテーションをつけることでコンテンツに対する理解を深めることができるとともに、コンテンツの再利用性を高めることができる。

2.3.1　発言の関連付け

発言と発言の関係がどのようであるかを知ることができると、関係をたどっていくことにより議論の流れを知ったり、議論が発散しているのか、それとも収束していっているのか、といったことが分かったりすることができるであろう。

本システムにおいては、発言を入力する際に、スライドと関連付けて入力し、さらに、「質問」「回答」「コメント」に分類して入力することにより、発言と発言の関係がある程度得られる。

さらに、それ以上の議論構造を得るために、本システムでは発言のグラフ表示・編集モードを設けた。

議論構造をグラフ状に表示することにより、議論内容の把握を容易にしたり、会議の場で有効な発言を促す効果があると思われる。

本システムにおいては SVG (Scalable Vector Graphics) [3] を用いた発言のグラフ表示・編集モード (図 2.3) を設け、議事録の構造を可視化した。発言とスライドの関連情報やキーワードなどを用いて半自動で構造化して表示し、それに対してユーザが編集を行うことが可能である。

グラフ表示モードは発言の関係をグラフィカルに表示することで議論の流れを可視化する。

このモードはSVGの中にJavaScriptを埋め込んで、SVGとJavaScriptを連携させることでこの画面で発言の関係を編集することが可能である。

発言一つ一つは四角形で囲まれて表示されており、それぞれに、その発言が「質問」であるか「回答」であるか「コメント」であるか分かるようにラベルがついている。

一つの発言をドラッグし、もう一つの発言にドロップする、という直感的な作業で発言を関連付けることができる。関連付けられた発言は矢印付きの線で結ばれる。

関連付けを解除したいときは、関連付けられた発言同士の間にできている線上で右クリックでててくるコンテキストメニューから「関連付けを解除」を選択することで関連付けを解除し、線を消すことができる。

発言の位置と関係は「Save」ボタンを押すか、右クリックでててくるコンテキストメニューから「議事構造の保存」を選択することで保存され、データベースが更新される。

2.4 データベースへの登録

入力されたテキストとスライドから得られたテキスト情報などをあわせて議事録が作成される。

作成される議事録はXML形式であり、以下のような構造になっている。

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<seminarInfo key="この議事録の識別子">
 <presenterID>議題提供者の ID</presenterID>
 <date>議論が行われた日付</date>
 <slideInfo>
  <file>使用するスライドのファイル名</file>
  <title>議題</title>
  <slide num="1">
   <title>一枚目のスライドのタイトル</title>
   <text>一枚目のスライドに含まれるテキスト</text>
  </slide>
  <slide num="2">
   <title>二枚目のスライドのタイトル</title>
   <text>二枚目のスライドに含まれるテキスト</text>
  </slide>
        :
        :

 </slideInfo>
 <presenterInfo>
  <name>議題提供者の名前 (ローマ字)</name>
  <名前>議題提供者の名前 (漢字)</名前>
  <affiliation>所属している組織</affiliation>
```

```
  <email>メールアドレス</email>
  <url>ホームページ</url>
 </presenterInfo>
 <minute>
  <authorID>書記の ID</authorID>
  <discussion>
   <item plane="0">一枚目のスライド以前に行われた議論の内容</item>
   <item slide="1">一枚目のスライドの時に行われた議論の内容</item>
       :
       :

   <item slide="3">
    <question id="発言の ID" x="グラフ表示した際の x 座標" y="同じく y 座標">発言
内容 (質問)</question>
    <answer id=" " x=" " y=" " relateTo="関連のある発言の ID">発言内容 (回答)</answer>
   </item>
   <item slide="3">
    <question id=" " >発言内容 (質問)</question>
    <answer id=" " >発言内容 (回答)</answer>
    <comment id=" " >発言内容 (コメント)</comment>
   </item>
       :
       :

  </discussion>
  <participant>
   <item>
    <name>参加者の名前 1 </name>
   </item>
   <item>
    <name>参加者の名前 2 </name>
   </item>
       :
       :

  </participant>
  <note>書記のメモ</note>
  <reference>
   <item>
    <url>参照した URL</url>
    <description>URL の説明</description>
```

```
   </item>
   <item>
    <url> </url>
    <description> </description>
   </item>
   <item>
    <file>参照したファイル</file>
    <description>ファイルの説明</description>
   </item>
       :
       :

  </reference>
  <evaluation>
   <good>good の評価の数</good>
   <good_soso>good_soso の評価の数</good_soso>
   <soso>soso の評価の数</soso>
   <soso_bad>soso_bad の評価の数</soso_bad>
   <bad>bad の評価の数</bad>
  </evaluation>
 </minute>
</seminarInfo>
```

実際に作成された議事録 XML を図 2.4 に示す。図 2.5 は議事録 XML の discussion タグ以下の発言を記録する部分を展開して表示したものである。データベースに直接登録するため、実際にはファイルは作成されないが、ここでは例示するために作成したファイルを提示する。

作成された議事録 XML は、Java Servlet を用いてサーバマシンの XML データベースとリレーショナルデータベースに登録される。

本研究においては、XML データベースとして Xindice [4] を、リレーショナルデータベースとして PostgreSQL を用いた。議事録データを保持しておくだけなら Xindice だけでも十分なのだが、検索速度の遅さから、検索の速い PostgreSQL も併用して使用する。

Xindice では、XML 文書をファイルシステムに似たモデルで管理する。ファイルシステムであればディレクトリとファイルからなるが、Xindice では、ディレクトリに対応するものを「コレクション」、ファイルに対応するものを「ドキュメント」と呼ぶ。さらに、ファイル名に対応するものを「キー」と呼ぶ。

Xindice には minute という名前のコレクションが作られており、その中に議事録 XML を格納していく。キーとして、例えば「shimizu_030304」のように、議題提供者の ID と日付をアンダーバーでつなげたものを使用し、それによって議事録を識別する。

議事録データを PostgreSQL に登録する際には、後での検索を意識して、細かくフィールド分けされたテーブルに入れていく。テーブルの構造としては以下のようになる。

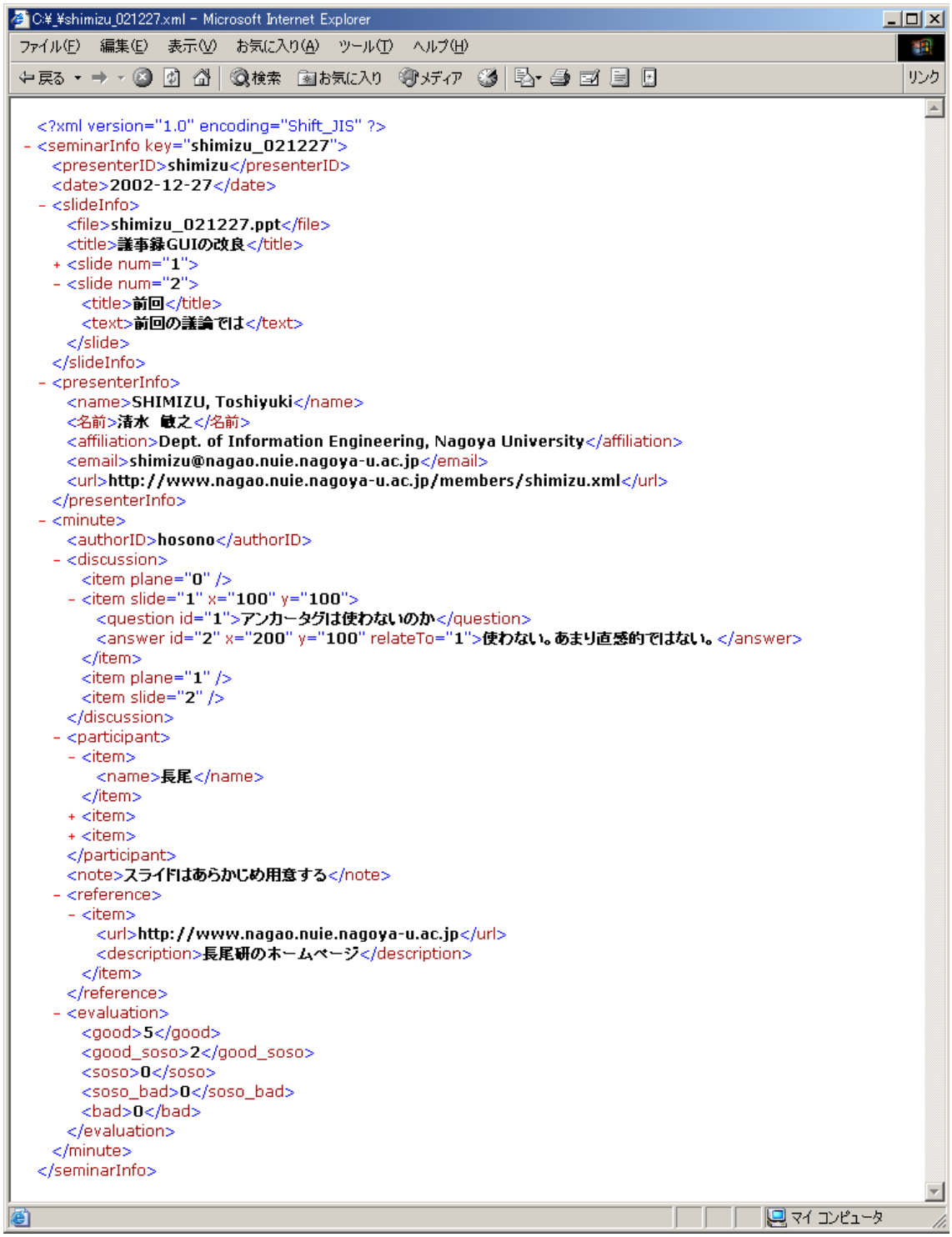
C:¥_¥shimizu_021227.xml - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り メディア リンク
<?xml version="1.0" encoding="Shift_JIS" ?>
<seminarInfo key="shimizu_021227">
<presenterID>shimizu</presenterID>
<date>2002-12-27</date>
<slideInfo>
<file>shimizu_021227.ppt</file>
<title>議事録GUIの改良</title>
<slide num="1">
<slide num="2">
<title>前回</title>
<text>前回の議論では</text>
</slide>
</slideInfo>
<presenterInfo>
<name>SHIMIZU, Toshiyuki</name>
<名前>清水　敏之</名前>
<affiliation>Dept. of Information Engineering, Nagoya University</affiliation>
<email>shimizu@nagao.nuie.nagoya-u.ac.jp</email>
<url>http://www.nagao.nuie.nagoya-u.ac.jp/members/shimizu.xml</url>
</presenterInfo>
<minute>
<authorID>hosono</authorID>
<discussion>
<item plane="0" />
<item slide="1" x="100" y="100">
<question id="1">アンカータグは使わないのか</question>
<answer id="2" x="200" y="100" relateTo="1">使わない。あまり直感的ではない。</answer>
</item>
<item plane="1" />
<item slide="2" />
</discussion>
<participant>
<item>
<name>長尾</name>
</item>
<item>
<item>
</participant>
<note>スライドはあらかじめ用意する</note>
<reference>
<item>
<url>http://www.nagao.nuie.nagoya-u.ac.jp</url>
<description>長尾研のホームページ</description>
</item>
</reference>
<evaluation>
<good>5</good>
<good_soso>2</good_soso>
<soso>0</soso>
<soso_bad>0</soso_bad>
<bad>0</bad>
</evaluation>
</minute>
</seminarInfo>
マイ コンピュータ

図 2.4: 議事録 XML

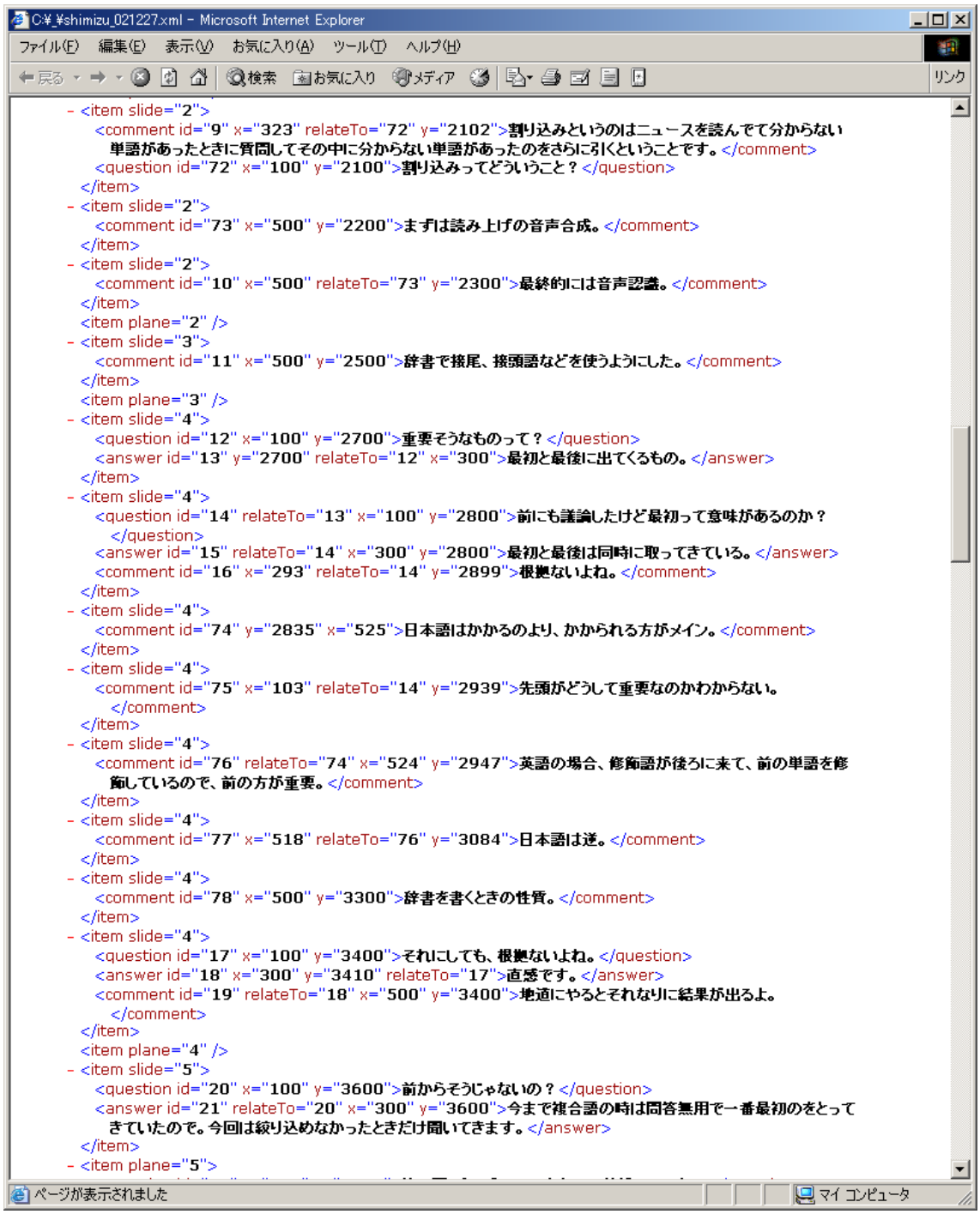
C:¥_¥shimizu_021227.xml - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
検索 お気に入り メディア
リンク
<item slide="2">
<comment id="9" x="323" relateTo="72" y="2102">割り込みというのはニュースを読んでて分からない単語があったときに質問してその中に分からない単語があったのをさらに引くということです。</comment>
<question id="72" x="100" y="2100">割り込みってどういうこと？</question>
</item>
<item slide="2">
<comment id="73" x="500" y="2200">まずは読み上げの音声合成。</comment>
</item>
<item slide="2">
<comment id="10" x="500" relateTo="73" y="2300">最終的には音声認識。</comment>
</item>
<item plane="2" />
<item slide="3">
<comment id="11" x="500" y="2500">辞書で接尾、接頭語などを使うようにした。</comment>
</item>
<item plane="3" />
<item slide="4">
<question id="12" x="100" y="2700">重要そうなものって？</question>
<answer id="13" y="2700" relateTo="12" x="300">最初と最後に出てくるもの。</answer>
</item>
<item slide="4">
<question id="14" relateTo="13" x="100" y="2800">前にも議論したけど最初って意味があるのか？</question>
<answer id="15" relateTo="14" x="300" y="2800">最初と最後は同時に取ってきている。</answer>
<comment id="16" x="293" relateTo="14" y="2899">根拠ないよね。</comment>
</item>
<item slide="4">
<comment id="74" y="2835" x="525">日本語はかかるのより、かかられる方がメイン。</comment>
</item>
<item slide="4">
<comment id="75" x="103" relateTo="14" y="2939">先頭がどうして重要なのかわからない。</comment>
</item>
<item slide="4">
<comment id="76" relateTo="74" x="524" y="2947">英語の場合、修飾語が後ろに来て、前の単語を修飾しているので、前の方が重要。</comment>
</item>
<item slide="4">
<comment id="77" x="518" relateTo="76" y="3084">日本語は逆。</comment>
</item>
<item slide="4">
<comment id="78" x="500" y="3300">辞書を書くときの性質。</comment>
</item>
<item slide="4">
<question id="17" x="100" y="3400">それにしても、根拠ないよね。</question>
<answer id="18" x="300" y="3410" relateTo="17">直感です。</answer>
<comment id="19" relateTo="18" x="500" y="3400">地道にやるとそれなりに結果が出るよ。</comment>
</item>
<item plane="4" />
<item slide="5">
<question id="20" x="100" y="3600">前からそうじゃないの？</question>
<answer id="21" relateTo="20" x="300" y="3600">今まで複合語の時は問答無用で一番最初のをとってきていたので。今回は絞り込めなかったときだけ聞いてきます。</answer>
</item>
<item plane="5">
ページが表示されました
マイ コンピュータ

図 2.5: 議事録 XML

カラム名	入れる内容
key	議事録の ID　Xindice におけるキーと同じ
title	議題
presenter	議題提供者の ID と名前
author	書記の ID
date	議論のなされた日付
presenterinfo	議題提供者の詳細情報
discussion	議論の内容全部
question	議論の内容のうち、「質問」のもの
answer	議論の内容のうち、「回答」のもの
comment	議論の内容のうち、「コメント」のもの
participant	参加者情報
note	書記のメモ
reference	参照 URL と参照 FILE の情報

このように細かくフィールド分けして、データを入れておくことにより、柔軟で高速な議事録の検索が実現できる。検索については次章で詳しく述べる。

第3章　議事録の再利用

生成された議事録は、議論中、または議論が終わった後、参照され、再利用される。

議事録 XML はサーバマシンのデータベースに蓄えられているが、サーバマシンを操作してデータベースから直接議事を見たり、それからファイルに変換して閲覧したりするのではあまりに不便である。

本システムでは、ブラウザ上からサーバマシンのデータベースにある議事録にアクセスして、閲覧することが可能である。

また、同じくブラウザ上からの操作によって、議事録を検索したり、要約したりすることが可能である。

これは、ユーザ側では、議事録を閲覧したり、検索したり、要約したりするのに特別なプログラムを必要とせず、さらに、インターネット上から議事録を扱うことができることを意味する。

3.1　議事録の取り出し

過去に行われた議論の議事録を取り出すには、大きく分けて 2 通りの方法がある。一つは議事録を一覧表示させてその中からユーザが選択する方法であり、もう一つはキーワードにより検索する方法である。

3.1.1　一覧表示から選択

過去に行われた議論の議事録を全て表示する。それぞれの議論のプレセンターと議論のなされた日付、ならびに議題の一覧がユーザに提示される。その中からユーザが取り出したい議事録を選択する。

図 3.1 に議事録の一覧表示の様子を示す。

3.1.2　キーワードによる検索

過去の議事録データのうち、必要とする議事録を Web 上の検索フォーム (図 3.2) から検索し、取り出すことが可能である。検索する際、議題、日付、発表者など、細かく検索フィールドが分かれており、目的とする議事録を容易に探し出し、取り出して閲覧することができるようになっている。検索部分ではデータベースとして PostgreSQL を利用するので高速性も期待できる。

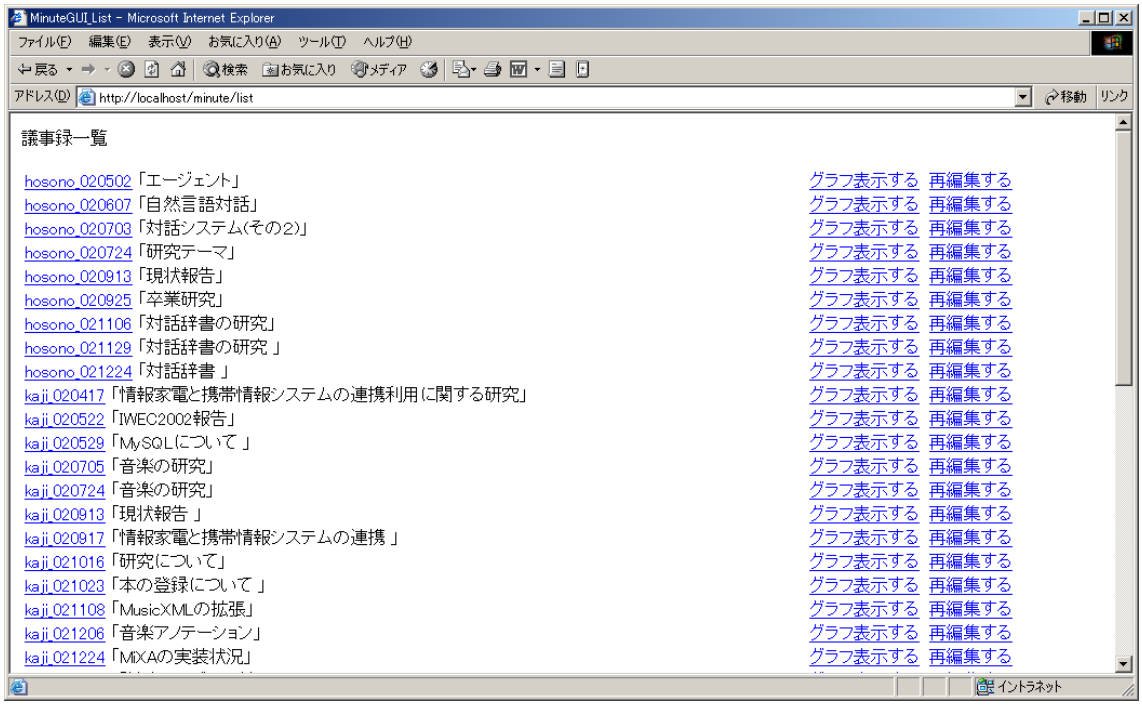

図 3.1: 議事録の一覧表示

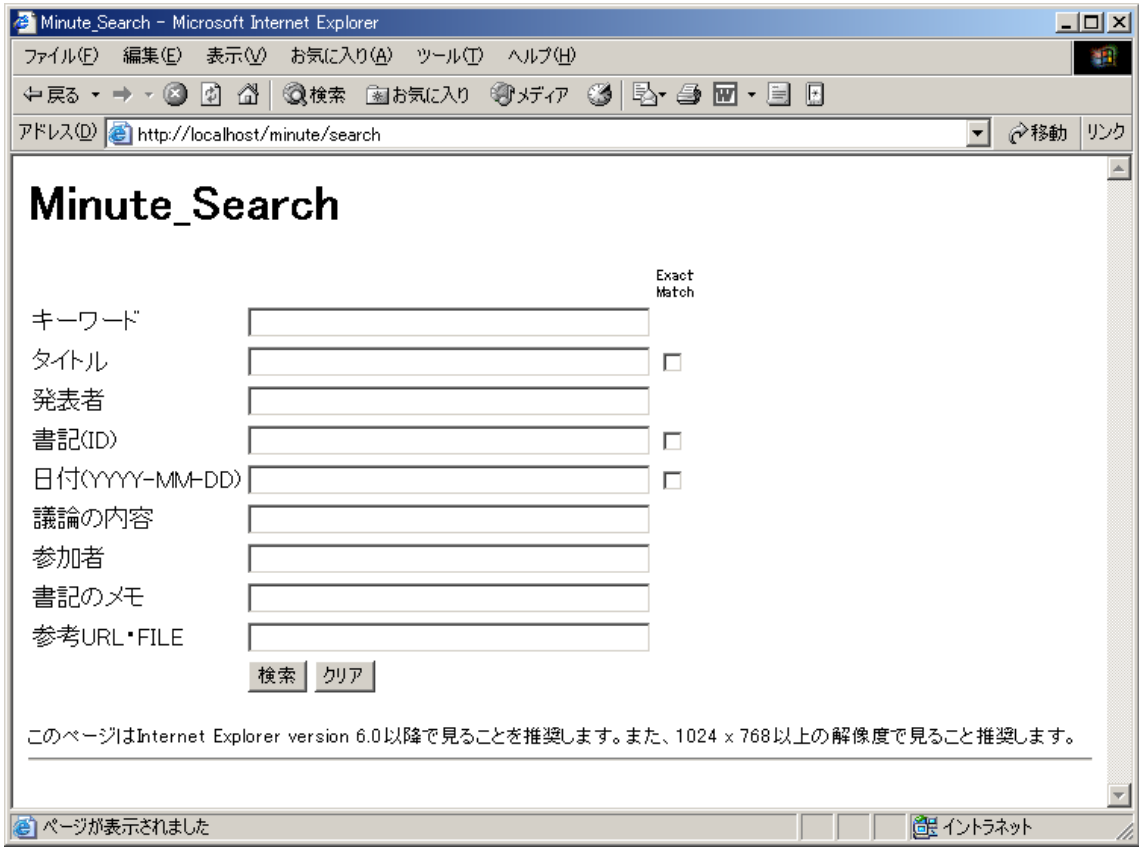

図 3.2: 議事録の検索フォーム

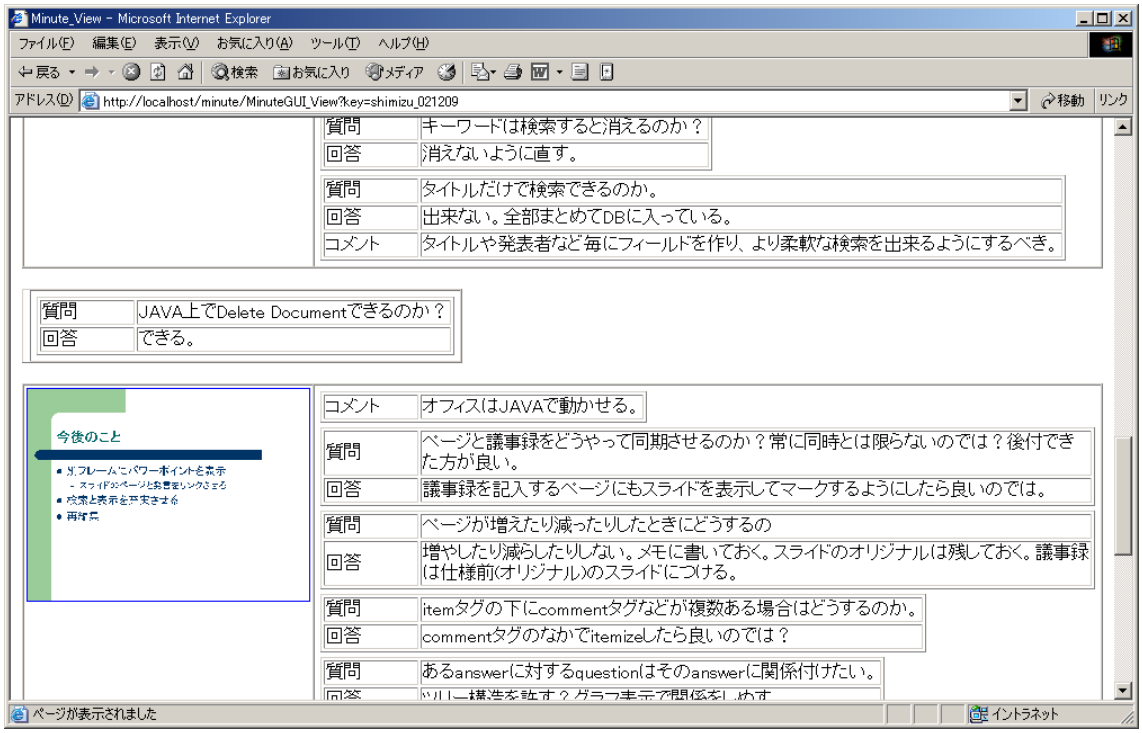

図 3.3: 時系列表示モード

検索結果は一覧表示と同様に、検索にヒットした議論のプレゼンターと議論のなされた日付、ならびに議題の一覧がユーザに提示される。

検索結果のページから議事録を閲覧すると、検索したキーワードがハイライトされて表示される。

3.2 議事録の閲覧

本システムにおいては、議事録を閲覧するにあたり、「時系列表示モード」と「グラフ表示モード」の 2 通りの表示モードを用意した。検索して取り出された議事録はこの 2 通りで閲覧することが可能である。

3.2.1 時系列表示モード

データベースに登録された議事録は、Xindice から取り出される DOM オブジェクトにスタイルシートをかけて表示することにより、ブラウザ上で閲覧することが可能である。

入力する段階で付けられたスライドとの関連情報を利用して、一つ一つの発言がどのスライドと関連付けられているかを表示した。図 3.3 に時系列表示モードでの表示の様子を示す。

現在進行中の議事録であっても、その途中までの進行状況を見ることが可能である。

3.2.2 グラフ表示モード

グラフ表示モードは発言の関係をグラフィカルに表示することで議論の流れを可視化する。

オンラインの掲示板などで行われる議論では、ツリー表示で議論の構造を表現しているものが多いが、オフラインで行われる会議の発言を構造化しようとする場合はそれでは不十分である。

ツリー表示ではなくグラフ表示にすることで、一つの発言から分岐していく話の流れの表現に加えて、複数の発言が一つの発言に収束される、といったことも表現できる。

時系列表示モードは純粋に見るだけのモードであるが、グラフ表示モードはある程度の議事録の編集も可能である。

このグラフ表示モードにより、議論中に有効な発言を促したり、後で見たときに議論内容の把握を容易にしたりする効果があると考えられる。

これは、前章 2.3 の図 2.3 で示したものである。

このグラフ表示モードで閲覧する場合も、書記の記入した発言はオンタイムで反映され、表示される。

3.3 議事録の再編集

検索して取り出された議事録は、やはりブラウザ上で再編集することも可能であり、新たに書き加えたり、入力ミスを訂正したりすることが可能である。

実際に議論が行われてからしばらくたって、何かアイデアがひらめいたり、その時に答えられなかった質問に対する回答ができるようになったりする、ということは十分にありえる状況であろう。

そのような時には、議事録を再編集して情報を付け加えることができる。

3.4 議事録の要約

議事録データを閲覧する際、要点だけを見たい場合は、アノテーションに基づいて計算された発言の重要度を用いて、要約して表示することも可能である。

本研究では要約するのに入力時に付けられた発言の構造と発言に含まれるキーワードを利用する。

発言一つ一つには固有の ID がふられており、それぞれの発言を識別することができる。そして、発言一つ一つに対して、重要度を計算し、重要度がより高いものを、指定された割合だけ表示する。重要度の計算はアノテーションに基づき、以下のように行われる。

グラフ表示モードで付けられた発言構造から、発言に付けられたリンクの数を計算し、リンクが多くついている発言は重要度が高いものとした。これは多くの発言を参照したり、多くの発言から参照されたりする発言は要点をついている可能性が高いことが予測されるからである。

また、発言内容を日本語形態素解析器である茶筌 [5] で解析し、解析結果が「名詞」または

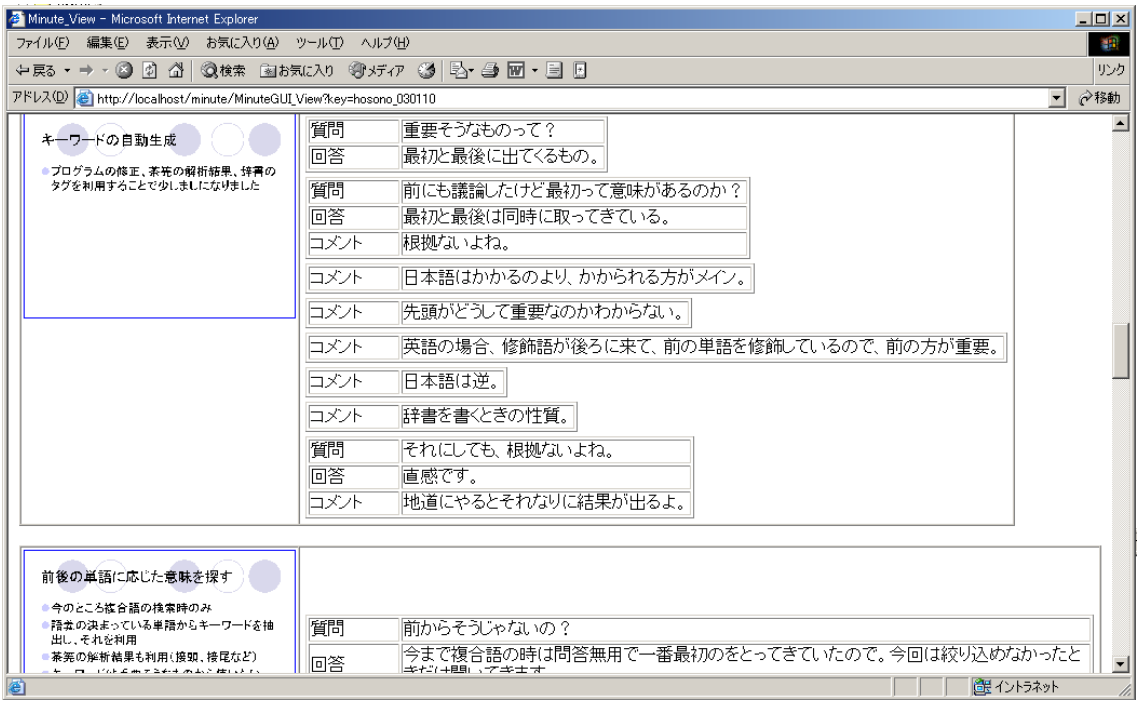

図 3.4: 要約前の議事録

「未知語」に分類される言葉をキーワードとしてそれを用いた発言の重要度付けも行った。

議題やスライドに含まれるテキストも同様に茶筌で解析し、それらに含まれているキーワードを発言も含んでいたら、その発言の重要度を上げた。そのとき、議題に含まれているキーワードを含んでいる発言の重要度をより高く上げ、それぞれのスライドのタイトルに含まれているキーワード、スライドのテキストに含まれているキーワードの順に高い重要度を発言に与えた。議題に含まれているキーワードを含んでいる発言は、その会議の本質的な話題についてのものであることが予想されるし、スライドのタイトルやスライドのテキストに含まれるキーワードも、それぞれそれなりの重要度があると考えられる。

また、発言全体の中でのキーワードの出現頻度も考慮し、何度もでてくるキーワードを含む発言の重要度を上げた。

以上から、本システムにおいて発言の重要度は、発言につけられたリンクの数による重要度 + 発言に含まれるキーワードによる重要度、で計算され、発言につけられたリンクの数による重要度は、単純にリンクの数に比例したものであり、発言に含まれるキーワードによる重要度は、議題やスライドのタイトル、スライドのテキストに含まれるキーワードなども考慮したキーワードの出現頻度によって決まるものである。

要約前の議事録の一部を図 3.4 に示し、本システムで要約した対応する部分を図 3.5 に示す。要約すると、関連付けられた発言のないスライドは縮小して表示される。

本システムで、要約にはグラフ表示・編集モードでのリンクのつき具合が大きくかかわってくるので、たとえ重要な発言でも、その発言についているリンクの数が少ないと要約したとき表示されない、という問題がある。

そのような発言が重要であることを自動で認識することは難しく、手動で重要度をあげてやることができるような仕組みを組み込むことを考えている。

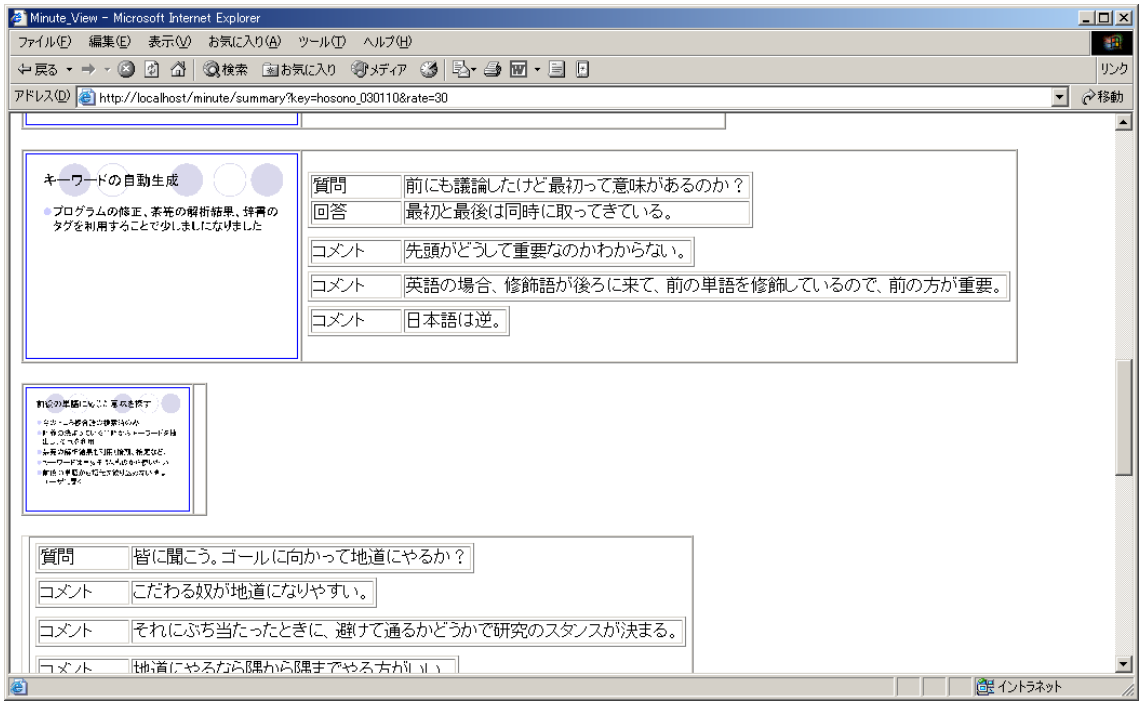

図 3.5: 要約後の議事録

また、キーワードによる発言の重要度も計算しており、出現頻度の高いキーワードを含む発言の重要度を上げているが、出現頻度の低いキーワードの重要性も一概に低いとは言い切れない。これに対しても同様のことが言える。

第4章　関連研究

ここでは会議支援などの関連研究を示す。議論を支援するグループウェアの代表的なものとして、gIBIS [6] があげられる。また、電子メールベースのグループウェアには The Coordinator [9] がある。議事録を生成するのにビデオデータなどを用いたものにミーティングキャプチャ[7] があげられる。オンライン上でのネットミーティングを支援するものには AIDE [8] や AVM [10] がある。それぞれについて簡単に説明する。

4.1　gIBIS

構造的アプローチによるグループウェアの代表的なものに gIBIS [6] があげられる。

これが使用している IBIS モデルでは発言の内容を issue(問題)、position(立場)、argument(意見)、other(その他) の４つの要素に分け、これらの発言間の関係を、賛成、反対、応答、一般化、特殊化、提案、支持、その他、で表現している。

利用者は IBIS モデルに基づく議論のプロトコルに従い、発言する際に、発言のタイプと結びつける発言のとの関係を明示する。このような議論展開を、グラフィカルなネットワーク表現で利用者に提供することで、議論の流れや状況の把握を促している。

4.2　The Coordinator

The Coordinator [9] は Speech-Act 理論を会話の理論 (Conversation Theory) へと発展させた Winograd 等のモデルに基づく電子メールベースのグループウェアである。

The Coordinator は利用者に明確なコミュニケーションのインタフェースを示し、送り手と受け手の認識にずれが生じないようにしている。利用者がメッセージを送信する際には、その状態に適したいくつかのメッセージ種別から選択していくため、システムは会話がどの状態にあるのかを用意に追跡できる。その結果システムは、利用者が抱えている依頼がどのような状態であるか、あるいは利用者の出した要求がどこまで処理されているのかという情報を提示することができる。

4.3　ミーティングキャプチャ

議論を再利用可能な形にするためにビデオ (映像と音) を主体として用いる手法もある。

ミーティングキャプチャ[7] は、講義、講演会、討論のような会議の内容を記録する。記録対象としては、ビデオや音声のほかに、講演のスライド、参加者によって書かれたメモ

といった会議の内容に踏み込んだデータや、講演者の身振り手振り、講演者の視線といった非言語的なデータも記録対象となる。

4.4　AIDE

AIDE(Augmented Informative Discussion Environment) [8] は電子化メディア上の対話を支援する環境を提供する。その機能としては、現時点の対話情報の意味的な構造を可視化して、参加者全員の共有情報空間を提供する Discussion Viewer、仮想的な対話参加者として、その時点でのディスカッション空間に関連するテキストを外部テキストベースから自動抽出し、それをディスカッション空間に投げ込んでくれる Conversationalist、対話に参加する各ユーザが、ディスカッション空間上の情報を複製・修正することで、共有情報を個人化することができる Personal Desktop の 3 つがある。

4.5　AVM

非同期型音声会議システム AVM [10] は非同期的な音声会議を効率的に実現するためのインタフェースである。

電子メールや電子掲示板などが持つ非同期型通信の利点を損なわずに、表現力が高く発言しやすい、という音声メッセージの利点を生かすため、オーバーラップ発話を利用し、クライアント側では文字表示によって発言の視覚化を行う。

第5章　おわりに

本章では、本研究のまとめと今後の課題について述べる。

5.1　まとめ

本論文では、会議における議論内容をデータベースに登録し、再利用するシステムについて述べた。

本研究で試作したシステムは、ブラウザ上から議事録の入力作業を行うことができる。その際、PowerPoint で作成されたスライドの情報を取り出し、スライドに含まれるテキストを議事録に追加するとともに、議事入力ページにスライドのイメージを表示し、発言とスライドを関連付けて入力することができる。また、SVG を用いて JavaScript と組み合わせることで、議論構造をグラフィカルに表示・編集することを可能にし、議事録にアノテーションをつけ、再利用を促進した。

アノテーションにより構造化された議事録は XML 形式で記述されており、XML データベースである Xindice とリレーショナルデータベースである PostgreSQL に登録される。登録された議事録は、検索することができ、取り出して閲覧できるほか、要約などの手段により再利用が可能である。

5.2　今後の課題

今後の課題として以下のものについて考察し、システムに反映していく予定である。

5.2.1　要約精度の向上

本システムにおける議事録の要約は、十分な実験・評価を行っておらず、その精度は十分ではないと考えられる。

議事録を特徴付けるようなキーワードの性質として、その議事録に数多く、つまり高い頻度で現れ、かつ、少ない数の議事録にしか現れない、という二つを考え、tfidf によるキーワードの重要度を計算し、そのような重要度を用いた議事録要約を考えている。

また、発言に対する重要度を手動で付けれるような仕組みを導入することにより、議事録にアノテーションをつけ、議事録の再利用性を上げるとともに、要約精度を向上させることを考えている。

5.2.2 発言の関連付けリンクの半自動生成

本システムでは発言のグラフ表示・編集モードを設けて、発言の関係をグラフィカルに提示したが、発言のリンク付けは、ほぼ手作業で行わなければならない。このリンク付けの作業を、発言に含まれるキーワードなどから半自動で行い、その後で人間が修正・追加できるようにする予定である。

5.2.3 複数の議事録にまたがる構造化

本研究で試作したシステムにおいて作成される議事録は、特定の会議に閉じたものであり、過去の会議との関連性は明示的には入っていない。キーワードなどから半自動的に過去の議事録との関連性を調べ、複数の議事録にまたがった要約などができるようにすべきであろう。

5.2.4 音声の利用

音声を利用することにより、議事録に載せられなかった情報も、後で参照できるようになると思われる。音声はそのままでは検索しにくい情報であるが、音声認識を用いてトランスクリプトを作成するなど、アノテーションを用いることにより、検索等の問題はかなり解消される。

5.2.5 画像の利用

画像情報があると、その場の雰囲気など、他の情報からは得にくいデータも利用できるようになると思われる。

謝辞

本研究を行うにあたり、日頃御指導を賜わった長尾確教授には研究の基礎的な考え方から論文指導など様々な面でお世話になりました。まず、御礼申し上げます。

梶克彦先輩には JACOB の利用法など、貴重なご指導を頂きました。山根隼人先輩には積極的に議事録入力に協力して頂きました。また松浦真治君にはデータベースに関して、大変お世話になりました。山本大介君、加藤範彦君、細野祥代さんには研究に関し、貴重な意見を頂きました。ありがとうございます。

長尾研究室秘書の兼松英代さんには学生生活ならびに研究活動のための様々なサポートをいただきました。また、長尾研究室の皆様には本研究を進めるにあたり有益な議論と様々な励ましを頂きました。この場を借りて御礼申し上げます。

関連図書

[1] W3C. Extensible Markup Language (XML). http://www.w3c.org/XML/. 1996.

[2] Dan Adler. The JACOB Project: A JAva-COM Bridge. http://danadler.com/jacob/. 2001.

[3] W3C. Scalable Vector Graphics 1.0 Specification. http://www.w3.org/TR/SVG/. 2001.

[4] The Apache XML Project. Apache Xindice. http://xml.apache.org/xindice/. 2001.

[5] 奈良先端科学技術大学院大学自然言語処理学講座. 形態素解析システム 茶筌. http://chasen.aist-nara.ac.jp/. 1997.

[6] Conklin, J. and Begeman, M.L.. gIBIS: A Hypertext Tool for Exploratory Policy Discussion. Proc. of CSCW '88. pp.140-152, 1988.

[7] 角康之, 間瀬健二. インタラクション・コーパス構築の試みとしてのミーティングキャプチャ. ヒューマンインタフェースシンポジウム 2002. 2002.

[8] 西本 一志, 門林 理恵子, 角 康之, 間瀬 健二, 中津 良平. マルチエージェントによるグループ思考支援. 電子情報通信学会論文誌, Vol.J80-D-I, No.5. pp.478-487, 1998.

[9] Winograd, T. and Flores, F.. Understanding Computers and Cognition. Addison-Wesley Publising Company, Inc. 1986.

[10] 西本卓也, 幸英浩, 川原毅彦, 荒木雅弘, 新美康永. 非同期型音声会議システム AVM の設計と評価. 電子情報通信学会論文誌 D-II, Vol.J83-D-II, No.11. pp.2490-2497, 2000.