音楽理論に基づくディスカッションマイニングのための議論構造エディタの開発

大島知之[†1]　浜中雅俊[†1, 2]　平田圭二[†3]　東条敏[†4]　長尾確[†5]

概要：本稿では音楽理論を用いて会議の議事録を構文解析し，発言の重要度を表現するタイムスパン木を生成するための議論構造エディタの開発について述べる．

Development of the Discussion Structure Editor for Discussion Mining based on Music Theory

Tomoyuki Oshima[†1]　Masatoshi Hamanaka[†1, 2]
Keiji Hirata[†3]　Satoshi Tojo[†4]　Katashi Nagao[†5]

Abstraction: In this paper, we explain the developing of the discussion structure editor, which generates time-span trees expressing degree of importance of statements.

1. はじめに

会議の内容を記録し，後に議事内容を要約や検索などの手段によって内容を参照することは非常に有益である．この時，会議の記録を容易に振り返ることができれば，その会議の決定事項や次回までの課題等を参加者が再確認でき，次回以降の会議の進行を効率良く行うことができる．

我々は会議を記録し，会議記録を自動要約することを目標として研究を行っている．会議記録の自動要約の課題を実現するためには，会議記録を分析して会議の流れや重要な発言を認識する必要がある．

そこで本稿では会議記録を分析し，会議の流れや重要発言を自動で同定するための第一歩として，手作業で議論構造を編集するためのエディタを提案する．提案するエディタでは会議の発言を時系列イベントとして捉え，各発言をノードとした木構造を作成することによって，発言同士の重要度の比較を可能する．

従来，発言を木構造化して表示するエディタとして，ディスカッションマイニングシステム[3, 4]のディスカッションメディアブラウザがあった．このブラウザでは発言をグラフのノードとし，ある発言とそれに継続した発言をリンクで結んだ木構造（ディスカッションマイニング木）として表示していた．これにより，各発言の関連性を視覚化することが可能であった．しかし，このブラウザは各発言の関連性の可視化を重視していたため，発言を時系列として捉えることが困難であった．本稿では，ある発言に対して継続する発言を一つの列に表示し，継続でない新たな発言ごとに列を追加することで，発言の関連性も示しながら時系列順でディスカッションマイニング木を表示する方法を提案する．提案するエディタでは，ディスカッションマイニングと共に，発言の重要度を表した木（タイムスパン木）を表示する．ディスカッションマイニング木を時系列順で表示することで，ユーザはディスカッションマイニング木とタイムスパン木の比較をすることが可能となる．

エディタ上に表示されるタイムスパン木は，発言数が多くなるほど複雑となり，編集に必要なステップ数は膨大となる．そのため，エディタに表示する編集前の木構造の初

†1 筑波大学
Tsukuba University
†2 JST さきがけ
PRESTO JST
†3 はこだて未来大学
Future University Hakodate
†4 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology
†5 名古屋大学
Nagoya University

期状態から目的の木構造を作成するまでにかかるステップ数をカウントし，どのような木構造を初期状態として表示すれば，より少ないステップ数で木の作成ができるか，実験を行った．実験の結果，議論の最初と最後の発言の木の高さを低くし，中央に近い発言ほど木の高さを高くするような，山形の木構造が，最も少ないステップ数で木の作成ができることがわかった．

2. 関連研究

本研究で提案する議論構造エディタは，ディスカッションマイニングシステムによって得られる議事録に対して音楽理論 GTTM を応用して構文解析を行い，各発言の重要度を木構造で表示するものである．本節ではディスカッションマイニングシステム，音楽理論 GTTM，音楽理論 GTTM を応用した会議記録の分析手法の３つの関連研究について説明する．

2.1 ディスカッションマイニングシステム

会議をマルチメディア議事録として記録するシステムとしてディスカッションマイニングシステムが提案されていた[4]（図 1）．マルチメディア議事録には従来の議事録のような発言のテキスト情報だけでなく，音声・映像や発表に用いられたスライド，これらを構造的に取り扱うためのメタデータを含んでいた．これにより，会議後に会議の様子を効率的に振り返ることができるほか，蓄積された議事録を自然言語処理によって類似した議事録からのまとめ検索や，議事録を時間順に並べることによって議論の発展を推測することを可能としていた．

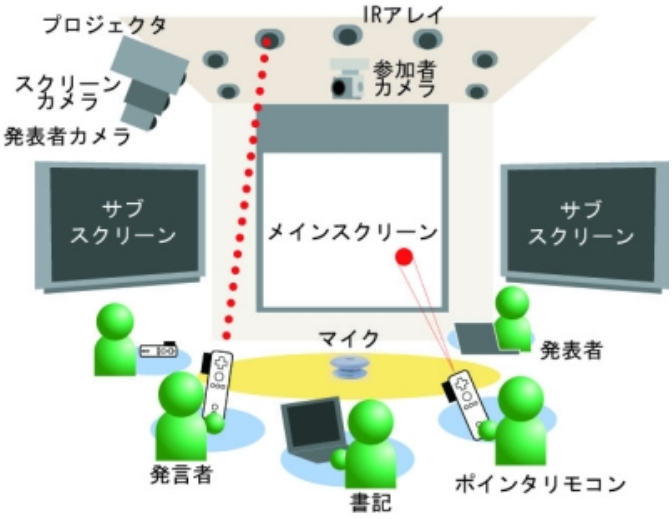

図 1　ディスカッションマイニングシステム

このシステムの対象としているディスカッションは研究室ゼミのような，スライドを利用した発表形式の会議である．発表に対する質問やコメントといった議論を行う際，発言者は発言の前に自分の発言が「導入」・「継続」のどちらかを選択する．導入はこれまでの話題から離れ，新しい話題の発言をする時に選択する．「継続」はこれまでの話題の流れに則り発言する時に選択する．例えば，発言者が発表の内容について質問をするときには導入を選択し，回答をする人物はその質問に対して継続することを選択する．これにより，ある発言がどの発言に対してなされたか，機械的な記録を実現していた．

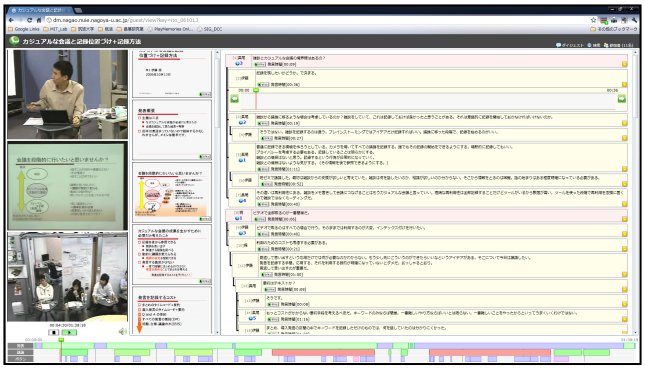

図 2　議事録コンテンツ閲覧画面

マルチメディア議事録の閲覧画面を図２にしめす．図２の左１列目は上から発表者，スライド，参加者の映像である．左から２列目は発表に使われたスライドを表示する．中央から右の領域は議論での発言内容を表示する．各発言はノードとし，継続の関係にある発言をリンクで結んだ木構造として表示する．この木構造をディスカッションマイニング木と呼ぶ．

2.2 音楽理論 GTTM

音楽理論 Generative Theory of Tonal Music（GTTM）は楽曲分析手法の一つである．GTTM は楽曲のフレーズの中から重要な音を探す分析手法であった．

GTTM の分析結果は音の相対的な重要度を表現するタイムスパン木と呼ばれる木構造で表現される（図 3）．例えば，隣接する２つのうち，１番左の音の方がより重要な場合，右の音は左の音の木の枝として表示する．図３の例では，１番左の音と２番目の音では左の音の重要度が高い．左の 16 分音符のグループに比べて，次の８分音符の重要度が高い．

音楽の構造分析を対象としたタイムスパン木の作成・編集を行うエディタは[4]で提案されていた（図 4）．しかし，会議の議事録を対象としたものではないため，会議記録の構造化に必要な発言情報の表示や，発言の関連性の表示はできなかった．

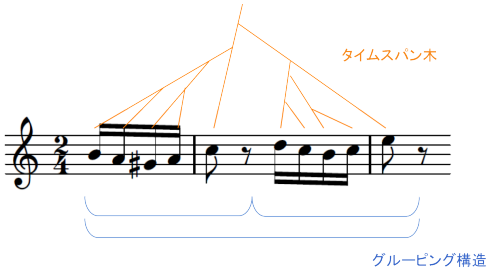

図 3　音楽分析でのタイムスパン木の作成

2.3 音楽理論を会議記録の分析に応用したディスカッションマイニング

本研究では，プレゼンテーション会議を発言の時系列とみなし，その時系列を音楽理論に倣って分析することで，議論を構造化することを目指している．

発言記録の構造化では GTTM の規則に倣って新たに作成した規則を適用する．例えば，時間的に近く，継続の関係にある発言同士はグループを作りやすい，明示的な参照や共通単語の多い発言同士はグループを作りやすい，発言時間が長いものや継続発言が多いものは重要な発言であることが多い，等である．これらの発言の情報を利用し，楽曲分析と同様，発言に関するタイムスパン木を生成する．

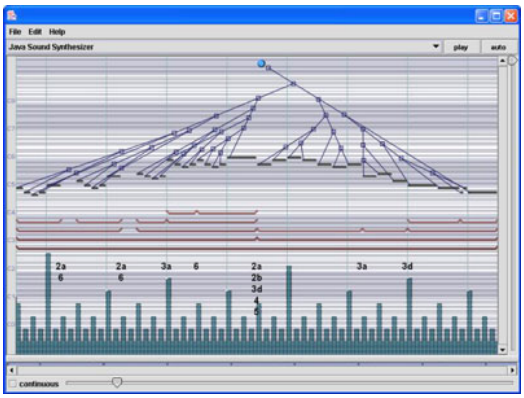

図 4　音楽の構文解析ためのツリーエディタ

2.4 エディタの位置づけ

議事録構造エディタを構築する目的は，議事録からタイムスパン木を作って，会議中の発言の重要度のランク付けをすることである．

ディスカッションマイニングシステムにおける議論構造エディタの位置付けを図 5 にしめす．図の左のようにスライドを用いた会議を想定する．参加者は個人用の端末（タブレット PC）を持参し，質問・回答・コメントなどで発言したい場合，端末から誰に対する発言かを選択してから発言する．また，発言した内容は発言者自ら端末に書き込む．これにより，システムのサーバに発言の情報が送られる．

この発言の情報は XML ファイルに書き出される．ここではこの XML ファイルを議事録 XML ファイルと呼ぶ．議事録 XML ファイルには各発言につき，発言番号・発言者名・発言時間・発言内容・誰に対する発言か，といった情報が含まれる．

議論構造エディタはこの議事録 XML ファイルを入力として，発言の情報を表示させる．このとき，誰に対する発言か，という情報をまとめることによって，発言の対応関係を木構造（ディスカッションマイニング木）として表現する．

会議中においては，DM システムによってリアルタイムに取得された発言を，エディタに入力，サブディスプレイや参加者の手元の端末にタイムスパン木を表示させる．タイムスパン木を会議中に閲覧できる利点としては，過去の発言のうち質問・回答・コメントといった会話が長く続いた際に，どの発言が重要なのか，これから発言しようとする内容と似た内容の発言が以前に出ていないか，といった確認を可能とする．

また，会議中に参加者個人の端末からツリー構造を編集可能にすることによって，エディタでのタイムスパン木作成にミスがあった場合，会議参加者側から木の訂正をすることを可能とする．

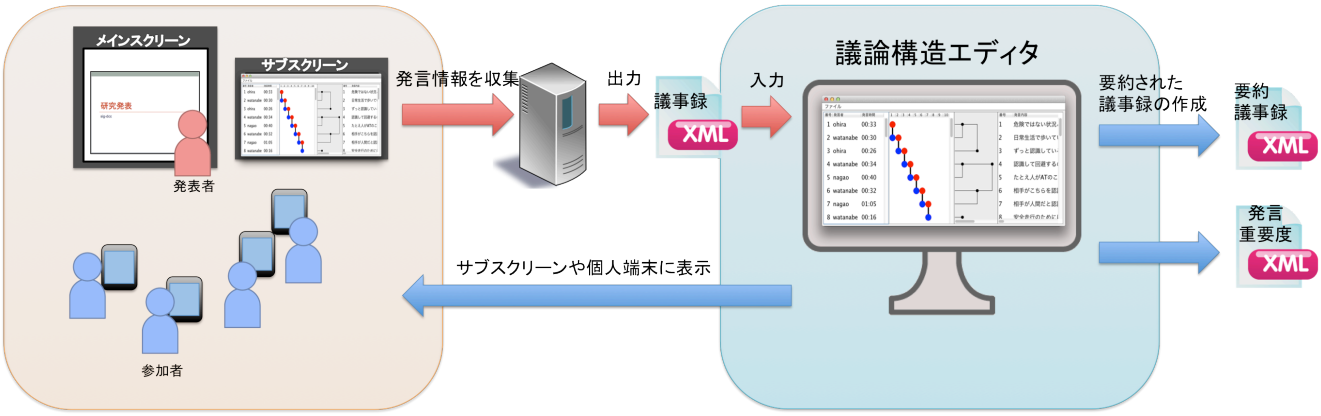

図 5　ディスカッションマイニングシステムと議論構造エディタとの関連性

会議終了後においては，各発言の重要度を木構造から読み取ることができるので，議事録とともに発言の重要度を記録しておくことが可能である．また，この重要度の情報があれば議事録を要約することもできるため，要約された議事録を出力することも可能とする．

2.5 エディタの要件

議論構造エディタに求められる機能と、それに対する設計について述べる。このエディタに求められる主な機能としては

① タイムスパン木の作成・編集はグラフィカルに行うことができるようにする
② 発言の内容や時間といった発言の情報を表示させながら、タイムスパン木の編集を行えるようにする
③ 会議中にもタイムスパン木の編集が行えるようにするために、新たな発言があった場合、その発言がエディタにも反映されるようにする。

の 3 点が挙げられる。

それぞれの要件に対する設計として、
①に対しては、Java2 SE の GUI コンポーネントである Swing を用いて設計し、マウスのドラッグアンドドロップ操作でタイムスパン木を編集することを可能にした。
②に対しては、ディスカッションマイニングシステムから出力された議事録 XML ファイルを入力とし、タイムスパン木の作成に必要な情報をエディタに表示するようにした。
③に対しては、会議中は常に入力された議事録 XML ファイルを更新するようにし､新たな発言がなされた場合でも、その発言がエディタに表示されるように設計した。

3. 議論構造エディタの表示方法

本章では議論構造エディタの表示について述べる．
議論構造エディタの例を図 6 に示す．
議論構造エディタは 4 つの領域から成り，図の左から順に，
・発言者表示領域
・ディスカッションマイニング木表示領域
・タイムスパン木表示領域
・発言内容表示領域
となっている．
全ての領域は上から時系列順に発言が並んでいる．例えば，発言番号 34，"watanabe"の発言についての木構造の情報が知りたいときには，横方向に見ることによって，その発言の木構造，発言内容を知ることができる．
ディスカッションマイニング木とタイムスパン木の表示方法とその意味について述べる．

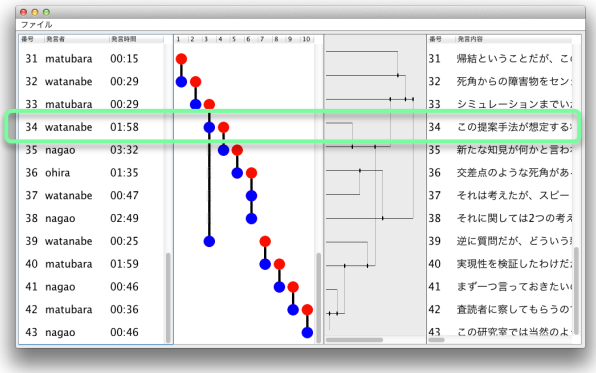

図 6　議論構造エディタ

3.1 ディスカッションマイニング木

ディスカッションマイニングシステムを用いて会議中に発言をする場合，発言者は自分の発言が「導入発言」と「継続発言」のどちらかであるかを宣言する．今まで続いていた話題から離れて発言する場合，その発言は導入発言となる．反対に，先行する発言に対して質問するなど，話題が続く場合には継続発言となる．継続発言をする際は，どの発言に対する発言なのかを指定する．

ディスカッションマイニング木は，発言の継続関係を木構造によって表したものである．その例を図 7 に示す．図 7 では<1>の導入発言に<2>と<3>の発言が継続している．また，<2>の発言には<4>が，<3>の発言には<5>がそれぞれ継続している．

図 7 のようにディスカッションマイニング木を表すと，発言の継続関係は明白になる．しかし，議論構造エディタにおいてはディスカッションマイニング木とタイムスパン木の比較をできるようにするため，上から発言番号順に並べて描く必要がある．

そこで我々は，発言が時系列順にならび，かつ発言の継続関係がわかる図 8 のような表示方法を提案する．ここで，図 7 と図 8 は同じ木構造を表している．図 7 の<1>の発言をみると，<2>，<3>の発言を子として持つことがわかる．図 8 では<1>の点から<2>，<3>の点が繋がるように線を引く．同様に図 8 の<2>には<4>の発言が継続するが，図 8 の<2>の点から<4>の点に繋がるように線を引く．

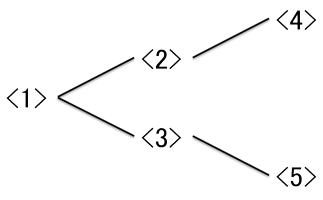

図 7　従来的な DM 木表示方法

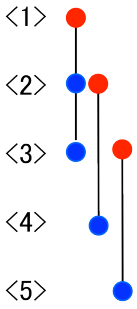

図 8
エディタ上での
DM 木表示方法

3.2 タイムスパン木

議論構造エディタにおけるタイムスパン木の表示の例を図９に示す．タイムスパン木は，ディスカッションマイニング木の表示と同様に，エディタの上から発言番号順に並ぶよう表示する．

タイムスパン木の編集方法について図 9 を 用いて説明する．図 9 の左では<1>の発言は<2>の発言から分岐している．<1>の発言が<3>の発言から分岐するようにしたい場合，<1>と<2>の交点を<3>の枝までドラッグアンドドロップすることによって，タイムスパン木を編集することができる．

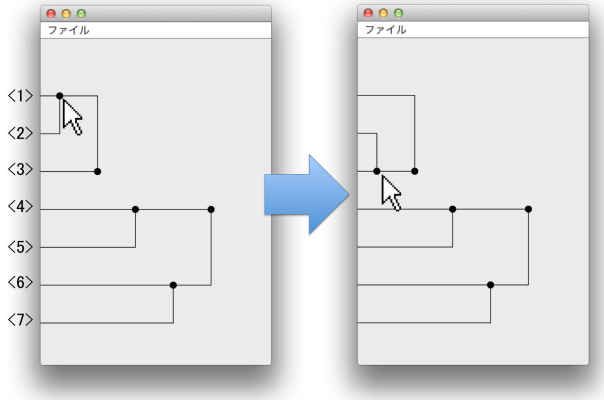

図 9　ドラッグアンドドロップでタイムスパン木を編集

3.3 タイムスパン木初期状態の作成

エディタを用いて議論のタイムスパン木を作成する際，次の手順を踏む．

① 議事録 XML ファイルを読み込む
② エディタに設定されている，タイムスパン木の初期状態の木構造を表示させる
③ 初期状態の木構造から，タイムスパン木作成のルールに従って木構造を編集する．

このときエディタがユーザに提示するタイムスパン木の初期状態の構造によって，タイムスパン木を完成させるまでの操作のステップ数が大きく異なるが予想される．タイムスパン木の作成を効率良く行えるようにするためには，初期状態から少ないステップ数でタイムスパン木を作成できるように，最適な初期状態を選出する必要があると考えられる．

4. 実験

タイムスパン木作成のため，エディタに表示する木構造の初期状態について実験を行う．木構造の初期状態からタイムスパン木を完成させるまでに，最も操作回数が少なかったものが良い初期状態であると考えられる．

4.1 実験方法

初期状態は次の３種類を用意した．

① 最初の発言の重要度を最も高く，後の発言になるにつれ重要度を低くした状態（図１０）
② ある話題における発言のうち，発言番号が中央である発言の重要度を最も高くし，その発言から離れるにつれ重要度を低く設定した状態（図１１）
③ 最後の発言の重要度を最も高く，前の発言になるほど重要度を低く設定した状態（図１２）

タイムスパン木を作る際に，これら３種類の初期状態それぞれから作成を開始し，タイムスパン木が完成するまでの操作ステップ数をカウントする．

実験に用いる議事録には，話題の数が８であるため，比較のために作成するタイムスパン木は８種類である．また，各話題に含まれる発言数はそれぞれ異なっており，最小で２，最大で１２である．そのため，話題ごとに必要となるステップ数が大きく変わると考えられるため，実験結果として得られたステップ数は正規化する．つまり，各話題において，ステップ数は０から１までの数で表される．総合的な結果として，各話題での正規化されたステップ数の平均を取り，一番値が小さかったものが最適な初期状態であると考えられる．

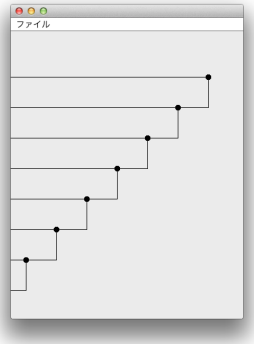

図 10　初期状態①

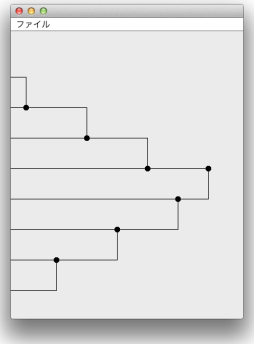

図 11　初期状態②

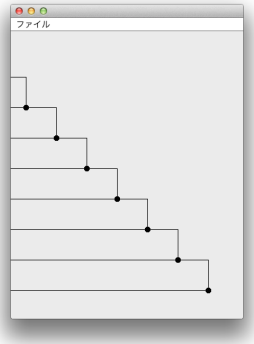

図 12　初期状態③

4.2 結果

実験の結果，表 1 のように，図 11 のように発言番号が中央である発言の重要度を高くした初期状態が，最もステップ数を少なくなった．このことから議論構造エディタに表示する木構造の初期状態として，図 11 のように中央を高くした木が最適だと考えられる．

表 1. 初期状態と正規化ステップ数

	正規化されたステップ数
初期状態①（図 10）	0.26
初期状態②（図 11）	0.14
初期状態③（図 12）	0.74

③の値が大きくなってしまった理由として，今回実験に使用した議事録データでは，話題の中で重要な発言が，話題の中心部や，前の方に集中することが多くあったため，③の初期状態だとステップ数が増えたと考えられる．

5. まとめ

本稿では，音楽理論を応用して会議の記録を分析する議論構造エディタについて述べた．従来，発言同士の継続関係を表すディスカッションマイニング木は時系列順での表示が困難であったが，本エディタでは，ある発言に継続する発言を一列に表示し，継続でない新たな話題の発言ごとに列を追加することによって，ディスカッションマイニング木を時系列順に表すことを可能にした．これによって，発言内容とディスカッションマイニング木，タイムスパン木を時系列順に表示することを可能とした．

また，タイムスパン木の最適な初期状態について検討を行い，実験の結果，発言番号の中央となる発言の木の高さを最も高くした山形の木構造を初期状態として表示させると，目的のタイムスパン木に至るまでの操作ステップ数を最も少なくなることがわかった．

今後，議論構造の自動分析に向けて，タイムスパン木作成のルール（グルーピング規則，ヘッド選択規則）の定式化，実装を行っていく．

参考文献

1) 平田圭二，長尾確，東条敏，浜中雅俊：音楽理論を会議記録の分析に応用したディスカッションマイニング，情報処理学会デジタルコンテンツクリエーション研究会，2012-DCC-1, No.16 (2012).

2) Lerdahl, F. and Jackendoff, R.: A Generative Theory of Tonal Music, The MIT Press (1983).

3) 土田貴裕，大平茂輝，長尾　確：対面式会議コンテンツの作成と議論中におけるメタデータの可視化，情報処理学会論文誌，Vol.51, No.2, pp.404-416 (2010).

4) Nagao, K., Kaji, K., Yamamoto, D. and Tomobe, H.: Discussion Mining: Annotation-Based Knowledge Discovery from Real World Activities, *Proc. of the Fifth Pacific-Rim Conference on Multimedia (PCM 2004)*, pp.522-531 (2004).

5) Masatoshi Hamanaka, Satoshi Tojo: Interactive Gttm Analyzer, Proceedings of the 10th International Conference on Music Information Retrievalconference (ISMIR2009), pp.291-296, October 2009.

付録
タイムスパン木 XML ファイルの構成

議論構造エディタ内部において，タイムスパン木の構造 XML ファイルにて記述される．例として，図１３のタイムスパン木 XML を記述する．

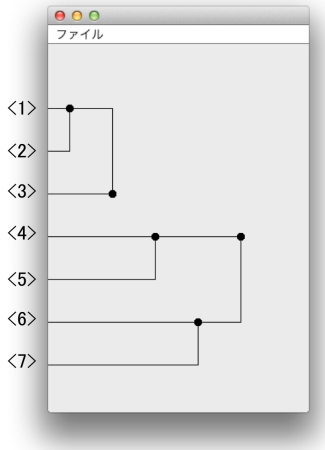

図 13　タイムスパン木

```
<?xml version="1.0"?>
<Document>
<ts type="root">
  <primary>
    <ts type="leaf">
      <statement id="3"/>
    </ts>
  </primary>
  <secondary>
    <ts type="inner">
      <primary>
        <ts type="leaf">
          <statement id="1"/>
        </ts>
      </primary>
      <secondary>
        <ts type="leaf">
          <statement id="2"/>
        </ts>
      </secondary>
    </ts>
  </secondary>
</ts>
```

```
<ts type="root">
  <primary>
    <ts type="inner">
      <primary>
        <ts type="leaf">
          <statement id="4"/>
        </ts>
      </primary>
      <secondary>
        <ts type="leaf">
          <statement id="5"/>
        </ts>
      </secondary>
    </ts>
  </primary>
  <secondary>
    <ts type="inner">
      <primary>
        <ts type="leaf">
          <statement id="6"/>
        </ts>
      </primary>
      <secondary>
        <ts type="leaf">
          <statement id="7"/>
        </ts>
      </secondary>
    </ts>
  </secondary>
</ts>
</Document>
```

タイムスパン木 XML ファイル中には primary と secondary のノードがある．これらは枝の分岐の部分に関して，より重要度の高い方を primary として記述している（図 14）．

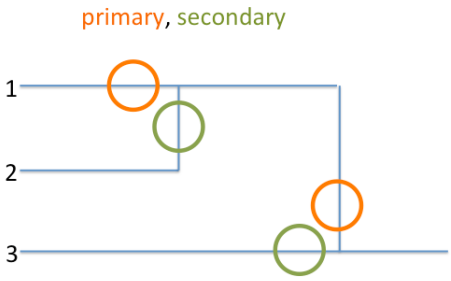

図 14　XML の primary, secondary ノードとタイムスパン木の対応関係