Internal Use Only（非公開）

TR－SLT－0077

テキスト音声合成ソフトウェア XIMERA マニュアル集
Collected Manuals of Text-to-Speech Software XIMERA

河野 みちよ　野見 隆志　長岡 直美
Michiyo Kono　Takashi Nomi　Naomi Nagaoka
速水 悦子
Etsuko Hayami
河井 恒　西澤 信行　国田 順子
Hisashi KAWAI　Nobuyuki NISHIZAWA　Junko KUNITA

2004 年 8 月 20 日

概要

本テクニカルレポートは，ATR の第 3 世代の音声合成ソフトウェア XIMERA(キメラ)のマニュアル集であり，「インストールマニュアル」，「コーパス情報」，「プログラマーズマニュアル」，「API リファレンス(抜粋)」，「ユーザーズマニュアル=XIMERA GUI=」，「ユーザーズマニュアル=データベース作成=」，「ユーザーズマニュアル=SAPI=」を収録している。ソフトウェアとしての XIMERA は，「音声認識・音声合成マルチクライアントプロジェクト」の一環として，音声言語コミュニケーション研究所と技術リエゾンセンタの共同作業により 2002 年度－2003 年度の 2 年間で開発された。本テクニカルレポートには，同プロジェクトの最終成果物(ver-1.1)に関する記録である。

（株）国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒619－0288「けいはんな学研都市」光台二丁目 2 番地 2 TEL：0774－95－1301

Advanced Telecommunications Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai “Keihanna Science City” 619-0288,Japan
Telephone:+81-774-95-1301
Fax :+81-774-95-1308

はじめに

本テクニカルレポートは，ATR の第 3 世代の音声合成ソフトウェア XIMERA(キメラ)に関するマニュアルまとめたものである。

ATR 設立以来現在までに 2 つの音声合成システムが開発されている。1 番目のシステムは，自動翻訳電話研究所(1986-1993)の研究成果として開発されたν-Talk であり，2 番目のシステムは，音声翻訳通信研究所(1993-2000) の研究成果として開発された CHATR である。XIMERA は，音声言語通信研究所(2000-2001)および音声言語コミュニケーション研究所(2001-2006)の研究成果として開発中のシステムであり，ν-talk から数えて第 3 世代目にあたる。

XIMERA は波形素片接続方式に基づいており，その点では CHATR と同様であるが，ソフトウェアとしては全く関連がない。一方で XIMERA においてはテキスト処理，音声コーパス，韻律予測，素片選択評価関数などさまざまな点で改良が行われており，音質面では格段に進化している。

XIMERA は，一般的な入力テキストに対する自然性の向上を第一の目標として開発されているため，多言語，話者性，発話スタイル，感情など合成音の多様化への配慮はされていない。このため，対象言語は，日本語と中国語のみであり，発話スタイルは基本的に朗読調のみを想定している。

ソフトウェアとしての XIMERA は，主として「音声認識・音声合成マルチクライアントプロジェクト」の一環として，音声言語コミュニケーション研究所(SLT)と技術リエゾンセンタ(TLC)の共同作業により 2002 年度－2003 年度の 2 年間で開発された。それぞれの担当範囲は，SLT がアルゴリズムとプロトタイプソフトウェアの開発，TLC がソフトウェアのコード整理，品質管理，および文書作成である。

XIMERA には，ソフトウェア開発の途上で作成された複数のバージョンが存在する。具体的には，ver-0.96，ver-0.97，ver-1.0，ver-1.1 の 4 つである。このうち ver-0.97 は開発用の内部向けバージョンであり，リリースはされていない(他のバージョンはすべてリリースされている)。「音声認識・音声合成マルチクライアントプロジェクト」における最終リリース版は，ver-1.1 である。開発版とリリース版の主な違いは，後者では中国語機能と日本語男声コーパスが取り除かれている点である。

本テクニカルレポートには，XIMERA-1.1 に関するマニュアルを収録している。収録したマニュアルの一覧を**表 1** に示す。「API リファレンス」は，オリジナルのページ数が 151 ページと多いため，C/C++ 用 API と JAVA 用 API の 2 系統の API が記述されている中で JAVA 用 API を削除し，さらに関数名と機能説明のみを抜粋し，収録した。「API リファレンス」以外の文書は，XIMERA-1.1 のリリースメディアに格納されているものと完全に同一である。

表 1. 収録したマニュアル・報告書の一覧

	タイトル	ページ数	主な内容
1	インストールマニュアル	1	外部ソース、インストール方法、ディレクトリ構成。
2	コーパス情報	7	配布コーパスの内容。
3	プログラマーズマニュアル	164	XIMERA エンジン、データベース作成、HTS 学習、SAPI 対応の処理概要、ファイルフォーマットなど。
4	API リファレンス(抜粋)	3	C/C++用 API50 個の概要。JAVA 用 API は省略。
5	ユーザーズマニュアル =XIMERA GUI=	71	JAVA サンプルプログラムの使用方法。
6	ユーザーズマニュアル =データベース作成=	77	データベース作成ツール，HTS 学習ツールの使用方法。音声切り出しツール，F0 集成ツール，サンプリングレート変換ツールなどの使用方法を含む。
7	ユーザーズマニュアル =SAPI=	11	Microsoft 社の SAPI で使用するための設定方法。
合計		334	

XIMERA（Ver1.1）

インストールマニュアル

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

「XIMERA」とは、ATR であらたに開発されたコーパスベース方式による音声波形素片接続型テキスト音声合成エンジン（開発コード名：XIMERA、以下「XIMERA」と記す）です。

XIMERA は、音声コーパスや音声データベース作成ツールなどとともに、音声合成研究やアプリケーション開発のためのシステムを、Linux や Windows 上で容易に構築できる環境を提供することを目的としたツール群を形成しています。XIMERA はその中核となる音声合成エンジンで、次のような特徴を持っています。

(1) 大規模な音声コーパスから選択された音声波形素片を接続する方式のため高品質な合成音声が得られる。
(2) 音声波形素片の選択接続に、知覚実験にもとづく評価関数を使用し自然性との対応性を向上させている。
(3) HMM 韻律合成により自然性の高い韻律制御を実現している。

本マニュアルは、XIMERA を開発するにあたっての技術に関する著作権及び使用条件、配布メディア、マニュアルの内容、インストール方法について説明します。

1．著作権及び使用条件について

XIMERA の開発にあたっては、NIH（Not Invented Here）にこだわらず技術導入することを基本方針にしています。
このため、表 1.1 に示す外部ソースを利用しています。

表 1．1 XIMERA Version1.1 で使用している外部ソース一覧

名称	Version	商用利用	再配布	使用箇所	主な機能
茶筌(ChaSen)	2.3.2	可	可	エンジン DB 作成	形態素解析
ipadic	2.6.2	可	可	エンジン DB 作成	IPA 品詞体系日本語辞書
南瓜 (CaboCha)	0.42	可	可	エンジン	係り受け解析
YamCha	0.27	可	可	エンジン	文節区切り解析
Darts	0.2	可	可	エンジン	Double-Array 用 Template Library
HTK	3.2.1	可	不可	DB 作成	隠れマルコフモデルツールキット
HTS	1.1.1	可	可	エンジン HTS 学習	韻律予測 HTS 学習
SPTK	3.0	可	可	HTS 学習	音声信号処理ツールキット
Pitch Tracker	1.0	可	可	DB 作成	F0 抽出

【茶筌】

茶筌システムは、広く自然言語処理研究に資するため無償ソフトウェアとして開発されたものです。茶筌の著作権は、奈良先端科学技術大学院大学情報科学研究科自然言語処理学講座（松本研究室）が保持します。本ソフトウェアの使用、改変、再配布については、特に制限はありませんが、再配布については、次の条件があります。

- 本ソフトウェアがその原形あるいは修正された形で再配布される場合は、再配布されるソフトウェアに茶筌 2.3 が使用されていることを明記すること。
- 再配布されるソフトウェアに、著作権に関する本節の記述と使用説明書の表紙裏ページの著作権に関するただし書きを必ず含むこと。

なお、本ソフトウェアの著作権者である奈良先端科学技術大学院大学は、原形あるいは改変された形で配布された本ソフトウェアに関連して生じる一切の損失に対して保証の責を負わないとしています。

詳細は、以下の URL をご覧ください。

http://chasen.aist-nara.ac.jp/

ユーザーズマニュアルは下記をご参照ください。

http://chasen.aist-nara.ac.jp/stable/doc/ipadic-2.6.2-j.pdf

【ipadic】

ipadic は、茶筌に付属している形態素辞書ファイルです。

ipadic の著作権は、奈良先端科学技術大学院大学情報科学研究科自然言語処理学講座（松本研究室）が保持します。

XIMERA では、ipadic にアクセント情報を追加し使用しています。

以下のサイトからダウンロード可能です。

http://chasen.aist-nara.ac.jp/stable/ipadic/

【南瓜】

CaboCha/南瓜は, Support Vector Machines にもとづく日本語係り受け解析器です。

CaboCha/南瓜の著作権は、奈良先端科学技術大学院大学情報科学研究科自然言語処理学講座（松本研究室）が保持します。

CaboCha は GNU Lesser General Public License (LGPL)に従ったフリーソフトウェアです。

詳細は、以下の URL をご覧ください。

http://cl.aist-nara.ac.jp/~taku-ku/software/cabocha/

【YamCha】

YamCha は、文節区切り解析器です。

YamCha の著作権は、奈良先端科学技術大学院大学情報科学研究科自然言語処理学講座（松本研究室）が保持します。

YamCha は GNU Lesser General Public License (LGPL)に従ったフリーソフトウェアです。

詳細は、以下の URL をご覧ください。

http://cl.aist-nara.ac.jp/~taku-ku/software/yamcha/

【Darts】

Darts は、 Double-Array を構築するための シンプルな C++ Template Library です。

Darts の著作権は、奈良先端科学技術大学院大学情報科学研究科自然言語処理学講座（松本研

究室）が保持します。
Darts は GNU Lesser General Public License (LGPL)に従ったフリーソフトウェアです。
詳細は、以下の URL をご覧ください。
http://cl.aist-nara.ac.jp/~taku-ku/software/darts/

【HTK】
The Hidden Markov Model Toolkit (HTK) is a portable toolkit for building and manipulating hidden Markov models. HTK is primarily used for speech recognition research although it has been used for numerous other applications including research into speech synthesis, character recognition and DNA sequencing. HTK is in use at hundreds of sites worldwide.
詳細は、以下の URL をご参照ください。
http://htk.eng.cam.ac.uk/

【HTS】
HTS とは、Hidden Markov Model Toolkit (HTK)、Speech Signal Processing Toolkit (SPTK) にパッチを当てた HMM-Based Speech Synthesis System (HTS)です。
HTS version1.1.1 は、フリーライセンスの下で公開されています。
詳細は、以下の URL をご参照ください。
http://hts.ics.nitech.ac.jp

【SPTK】
SPTK とは、東京工業大学、名古屋工業大学で開発された音声信号処理ツールキット（Speech Signal Processing Toolkit）です。
SPTK version 3.0 は、フリーライセンスの下で公開されています。
詳細は、以下の URL をご覧ください。
http://kt-lab.ics.nitech.ac.jp/~tokuda/SPTK/index-j.html

【Pitch Tracker】
Pitch Tracker は、ケンブリッジ大学で開発された F0 抽出プログラムです。
詳細は、以下の URL をご覧ください。
http://www-2.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/speech/systems/pt/0.html

２．Version1.0からの変更点

XIMERA Version1.1は、XIMERA Version1.0より次の点が変更になっています。

（１） 素片選択モジュール改良

素片選択のサブコスト関数を改良し、合成音品質を向上させました。
1音素に対するピッチ抽出を7点から5点に変更しました。

（２） 無声化ルール変更

無声化ルールを変更し、無声音を聞き取りやすくしました。

（３） アクセント改良

正しいアクセントが付与されるよう、辞書、プログラムを改良しました。

（４） データベース作成

ケプストラム配列数を40から24に変更しました。
また、素片選択モジュール改良に伴い、新たに追加になった２ファイルを作成します。

（５） HTS学習

学習に使用していたHTSのバージョンを、Ver1.1bよりVer1.1.1に変更しました。
また、入力ファイルである言語韻律情報ファイルの漢字コードがOS（Windows,Linux）に依存しなくなりました。

（６） SAPI対応

SILENCE、EMPHタグに対応しました。

（７） 音声切り出しツール(jspseg)機能追加

32bit 音声ファイル対応、文番号追加、マージン個別編集など、多数の機能を追加しました。詳細は、src¥dbtool¥jspseg¥whatsnew.txtをご参照ください。

（８） F0・ラベル修正ツール(piikun)制約事項追加

ラベル編集の際、音声区間を越えてラベルを設定できないようにするなど、制約事項を追加しました。詳細は、src¥dbtool¥piikun¥whatsnew.txtをご参照ください。

（９） XIMERA GUI機能追加

音声波形、スペクトラム表示画面にスクロール機能追加、話速・ピッチ設定での範囲

チェックなどを行うようにしました。

詳細は、src¥sample¥ximera-gui¥whatsnew.txt をご参照ください。

３．制限事項

XIMERA Version1.1には、機能上の制限事項があります。

・SAPI対応

XIMERAは、Microsoft Speech API（以下SAPI）に準拠しています。
SAPIでは、XML形式の記述により、ボリューム調整やピッチ調整など細かな制御を行うことができます。
XIMERA Version1.1では、PITCH、RATEの２つのタグに対応しておりません。
詳細は、「ユーザーズマニュアル =SAPI= 」をご参照ください。

4．データベースの互換性について

XIMERA Version1.0 で作成されたデータベースと、XIMERA Version1.1 で作成されたデータベースの互換性について説明します。

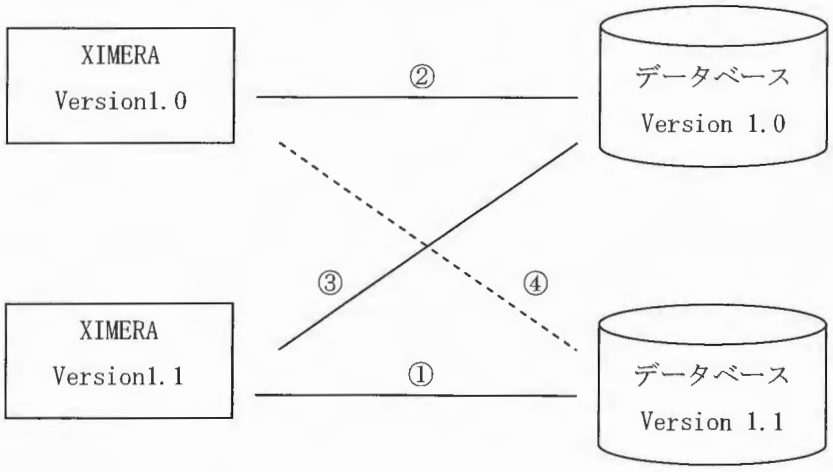


図 4.1 データベースの互換性

エンジンとデータベースの組み合わせと使用可否は次のとおりです。
（番号は図 4.1 内の番号に対応）

① 使用可能。
② 使用可能。
③ 使用可能。
④ 使用不可。データベースを Version1.0 で作成してください。

5．配布メディア

XIMERA Version1.1 のメディア構成は、表 5.1 XIMERA Version1.1 配布メディア一覧 のとおりです。
Windows、Linux 共通のメディアです。

表 5．1 XIMERA Version1.1 配布メディア一覧

配布メディア名	DVD-R/CD-R	枚数
XIMERA Version1.1	CD-R	1 枚
データベース・HTS 学習結果 Version1.1	DVD-R	1 枚
合計		2 枚

(1) XIMERA Version1.1 (CD-R)

XIMERA エンジンのソースと実行体(HTK 以外)を含みます。
また、辞書ファイル、各種サンプル、ドキュメントを含みます。
次のディレクトリに存在するデータです。

Ximera110/setupbin
Ximera110/bin
Ximera110/doc
Ximera110/sample
Ximera110/src
Ximera110/xdata/settings

メディア内のテキストファイルは漢字コード SJIS で記述されています。
Linux にインストールされる場合は、漢字変換スクリプトを使用し、漢字コード EUC に変換します。詳細は、「8. インストール」の章をご参照ください。

(2) データベース・HTS 学習結果 (DVD-R)

(2-1)データベース

データベースは、F009、F009_503 の 2 つで、以下のディレクトリに存在します。

Ximera110/xdata/F009/seg/F009　　(正味 47.6 時間)
Ximera110/xdata/F009/seg/F009_503　　(正味 35 分)

※　正味時間とは、1 文単位で、発声前後の無音を省いた時間です。
※　声質チェックデータが除外されているため 50 時間より少なくなっています。

コーパス(xdata/F009/corpus)以下は変更ありません。Version1.0 で配布したコーパスをご使用

ください。

(2-2) HTS 学習結果

HTS 学習結果は、XIMERA Version1.1 の HTS 学習ツールを用い、HTS 学習用コーパス（Version1.0 配布）より作成したものです。以下のディレクトリに存在します。

Ximera110/xdata/F009/hts/F009_hts

HTS 学習で使用している HTS モジュールのバージョンが Ver1.1b から Version1.1.1 に変更になっています。

HTS 学習結果は Ver1.0 と Ver1.1 で互換性があります。

6．XIMERA Version1.1動作環境

6.1 実行環境

ハードウェア

CPU：　Pentium4 2.0GHz以上
メモリ：　1GB以上
HDD：　15GB以上
　　ただし、48kHz音声を使用する場合は、45GB以上
光ドライブ：DVD-ROM、CD-ROM（読み取り装置）

ソフトウェア

OS：Windows XP、またはRed Hat Linux 7.2
J2SE™ v1.4.2_04

6.2 開発環境

ソフトウェア

OS：　Windows XP、Red Hat Linux 7.2
開発環境：　Windows上でのコンパイラとしてMicrosoft VisualStudio.NET
　　Linux上でのコンパイラとしてg++（version 2.96）
　　J2SE™ v1.4.2_04
　　Microsoft Speech SDK5.1
開発言語：　C++、Java

＜SAPIのインストールについて＞

WindowsXPにはSAPIが標準で組み込まれていますので、インストールの必要はありません。SAPI関連モジュールのコンパイルを行う場合や、SAPIのヘルプを参照する場合は、以下のサイトから入手し、インストールしてください。

http://www.microsoft.com/speech/download/sdk51/

7．マニュアルについて

XIMERA Version1.1 のマニュアルの構成は次のとおりです。

- インストールマニュアル
- コーパス情報
- プログラマーズマニュアル
  - コンパイル方法
  - システム概要
  - モジュール構成
  - 音声合成エンジン
  - AutoDBmake
  - HTS Training
  - XIMERA SAPI
- XIMERA API リファレンス
- ユーザーズマニュアル XIMERA GUI
- ユーザーズマニュアル DB 作成
  - 音声切り出しツール
  - サンプリングレート変換ツール
  - データベース作成ツール
  - F0・ラベル修正ツール
  - HTS 学習ツール
- ユーザーズマニュアル SAPI

※ 着色された枠単位で 1 冊のマニュアルとなります。

※ それ以外は、そのマニュアル内での主な項目を表します。

※ マニュアルは、配布メディア XIMERA Version1.1 CD-R の Ximera110/doc 以下にあります。

8．インストール

Windows、Linux 別にインストール方法を説明します。

8．1 インストール方法（Windows の場合）

Windows マシンにインストールする場合、以下の手順で行ってください。

(1) XIMERA エンジンのインストール

XIMERA Version1.1 CD-Rより、
Ximera110 以下のディレクトリを任意のディレクトリにコピーします。
XIMERA エンジンのインストールはこれで終わりです。
以降、任意のディレクトリを、インストールディレクトリと表現します。

(2) データベース・HTS 学習結果のインストール

インストール対象は以下のとおりです。

- データベース F009・F009_503
- HTS 学習結果 F009_hts

以下の手順でインストールを行います。
データベース・HTS学習結果DVD-R より、
Ximera110 以下のディレクトリをインストールディレクトリにコピーします。
（ディレクトリが重複しているため、上書き確認メッセージが出力されますが、すべて上書きで行ってください。）

・データベース

Ximera110¥xdata¥F009¥seg¥F009_503 以下
Ximera110¥xdata¥F009¥seg¥F009 以下

・HTS 学習結果（HMM）

Ximera110¥xdata¥F009¥hts¥F009_hts 以下

(3) 音声コーパスのインストール

音声コーパスはXIMERA Version1.0 と同様です。
以下のいずれかの方法でコーパスをインストールしてください。

方法１：XIMERA Version1.0 からコーパスを移動

XIMERA Version1.0 をインストール済みの場合のみ可能です。
Ximera100¥xdata¥F009¥corpus を
Ximera110¥xdata¥F009¥corpus に移動させてください。

XIMERA Version1.0 データベースの使用を継続される場合は、下記ファイルを編集してください。

Ximera100¥xdata¥F009¥seg¥F009¥data_wvc.txt
Ximera100¥xdata¥F009¥seg¥F009_503¥data_wvc.txt

data_wvc.txt 編集後サンプル

```
-f 16000
-shift 5.0
-f0shift 10.0
-wavdir ../../../../../Ximera110/xdata/F009/corpus/16k
-wavsign wav
```

網掛け部分に移動先コーパスのパスを設定してください。

方法２：XIMERA Version1.0 インストールメディアからコピー

XIMERA Version1.0 で配布しましたデータベース・HTS 学習結果 DVD-R3 枚より、
Ximera100¥xdata¥F009¥corpus¥16k 以下を、Ximear110¥xdata¥F009¥corpus¥16k にコピーしてください。
また、48kHz 音声コーパスをご使用の場合は、48kHz 音声コーパス DVD-R9 枚より、
Ximera100¥xdata¥F009¥corpus¥48k 以下を、Ximear110¥xdata¥F009¥corpus¥48k にコピーしてください。
詳細は、「XIMERA (Ver1.0) インストールマニュアル」をご参照ください。

(4) 48k 音声コーパス使用時の編集

リリースしたデータベースは 16kHz 音声ファイルを使用するよう設定されています。
48kHz 音声で合成させる場合のみ、データベースの下記ファイルを編集してください。

Ximera110¥xdata¥F009¥seg¥F009¥data_wvc.txt

Ximera110¥xdata¥F009¥seg¥F009_503¥data_wvc.txt

＜編集箇所＞

-f　16000 を 48000 に変更

-wavdir ../../corpus/16k を../../corpus/48k に変更

data_wvc.txt ファイル編集後サンプル

```
-f 48000
-shift 5.0
-f0shift 10.0
-wavdir ../../corpus/48k
-wavsign wav
```

網掛け部分が変更済み箇所です。

以上でインストールは完了です。
茶筌・南瓜の設定を行わないと XIMERA は正常に動作しません。
設定方法については、10 章「茶筌・南瓜設定」をご参照ください。

8．2 インストール方法（Linux の場合）

Linux マシンにインストールする場合、以下の手順で行ってください。

インストール方法は、以下のことを仮定して説明します。任意で指定した場合は、読み替えて作業してください。

インストールユーザー：ximera
インストールグループ：ximera
XIMERA Version1.1 のインストール先：/home/ximera/Ximera110
DVD-R のマウントポイント：/mnt/cdrom

(1) エンジンのインストール

XIMERA Version1.1 のCD-Rより、
Ximera110 以下のディレクトリを任意のディレクトリにコピーします。

配布メディアよりコピーを行う手順は以下のとおりです。
root でログインし、DVD-R をマウントします。

```
# mount /dev/cdrom /mnt/cdrom
```

DVD-R のデバイス名は、上記網掛け部分です。

ximera ユーザーでログインし、DVD-R の内容をコピーします。

```
$ cp  -rf  /mnt/cdrom/Ximera110  /home/ximera/
```

root ユーザーでアンマウントします。

```
# umount /mnt/cdrom
```

(2) データベース・HTS 学習結果のインストール

インストール対象は以下のとおりです。

- データベース F009・F009_503
- HTS 学習結果 F009_hts

以下の手順でインストールを行います。

データベース・HTS学習結果DVD-R をrootでマウントし、ximeraユーザーで、
Ximera110 以下のディレクトリをインストールディレクトリにコピーします。
コピーの前に書き込み権限の付与が必要です。

```
$ chmod u+w -R /home/ximera/Ximera110
$ cp  -rf  /mnt/cdrom/Ximera110  /home/ximera/
```

コピー完了後に、再度書き込み権限を付与します。

```
$ chmod u+w -R /home/ximera/Ximera110
```

DVD-R よりコピーが完了すると、以下がインストールされます。

・データベース

Ximera110/xdata/F009/seg/F009_503 以下

Ximera110/xdata/F009/seg/F009 以下

・HTS 学習結果 (HMM)

Ximera110/xdata/F009/hts/F009_hts 以下

(3) 音声コーパスのインストール

音声コーパスは XIMERA Version1.0 と同様です。
以下のいずれかの方法でコーパスをインストールしてください。

方法１ : XIMERA Ver1.0 へリンクを作成する

XIMERA Ver1.0 をインストール済みの場合のみ可能です。

```
[ximera]> ln -s Ximera100¥xdata¥F009¥corpus Ximera110¥xdata¥F009¥corpus
```

方法２ : XIMERA Version1.0 からコーパスを移動

XIMERA Ver1.0 をインストール済みの場合のみ可能です。

```
[ximera]> mv  Ximera110¥xdata¥F009¥corpus Ximera100¥xdata¥F009¥corpus
```

XIMERA Version1.0 データベースの使用を継続される場合は、下記ファイルを編集してください。

Ximera100¥xdata¥F009¥seg¥F009¥data_wvc.txt

Ximera100¥xdata¥F009¥seg¥F009_503¥data_wvc.txt

data_wvc.txt 編集後サンプル

```
-f 16000
-shift 5.0
-f0shift 10.0
-wavdir ../../../../../Ximera110/xdata/F009/corpus/16k
-wavsign wav
```

網掛け部分に移動先コーパスのパスを設定してください。

方法３：XIMERA Version1.0 インストールメディアからコピー

XIMERA Version1.0 で配布しましたデータベース・HTS 学習結果 DVD-R3 枚より、Ximera100/xdata/F009/corpus/16k 以下を、Ximear110/xdata/F009/corpus/16k にコピーしてください。

また、48kHz 音声コーパスをご使用の場合は、48kHz 音声コーパス DVD-R9 枚より、Ximera100/xdata/F009/corpus/48k 以下を、Ximear110/xdata/F009/corpus¥48k にコピーしてください。

詳細は、「XIMERA (Ver1.0) インストールマニュアル」をご参照ください。

(5) 48k 音声コーパス使用時の編集

リリースしたデータベースは 16kHz 音声ファイルを使用するよう設定されています。

48kHz 音声で合成させる場合のみ、データベースの下記ファイルを編集してください。

Ximera110/xdata/F009/seg/F009/data_wvc.txt

Ximera110/xdata/F009/seg/F009_503/data_wvc.txt

＜編集箇所＞

-f　16000 を 48000 に変更

-wavdir ../../corpus/16k を../../corpus/48k に変更

data_wvc.txt ファイル編集後サンプル

```
-f 48000
-shift 5.0
-f0shift 10.0
-wavdir ../../corpus/48k
-wavsign wav
```

網掛け部分が変更済み箇所です。

(6) 実行権限の付与

実行ファイルと、(6)以降で使用するシェルに対して権限を付与します。

ximera ユーザーで行います。

```
$ cd /home/ximera/Ximera110
$ chmod +x setupbin/chmod-exe.sh
$ setupbin/chmod-exe.sh
```

(7) 漢字コードの変換(ソースファイル)

配布メディア内のテキストファイルは SJIS で記述しています。

漢字コード変換ツールを利用して、ソースファイル、ヘッダーファイルの漢字コードを SJIS から EUC に変換します。

```
$ cd  /home/ximera/Ximera110/setupbin
$ ./sj2euc-ext.sh  /home/ximera/Ximera110/src
```

ただし、src/ximera-api/hankaku_kana.h については、Windows のみで使用するヘッダーファイルであり、ファイル内に半角カナが記述されているため、変換対象外としています。

(8) 漢字コードの変換 (言語韻律情報ファイル)

言語韻律情報ファイル(*.pli)の漢字コードを SJIS から EUC に変更します。言語韻律情報ファイルには、ファイル中に漢字コード名が記述されているタグがあります。SJIS から EUC に変更した際には、タグの内容も編集されます。ファイルを直接参照するなど、変換が必要な方は実行してください。

※　音声コーパスのインストールで、方法３（Ver1.0 のメディアからコピー）を実施された場合のみ実行してください。

```
$ ./sj2euc-pli.sh  /home/ximera/Ximera110/xdata/F009/corpus/hts16k
```

(9) 漢字コードの変換（カナファイル）

コーパス内のカナファイルも SJIS で記述されています。
ファイルを直接参照するなど、変換が必要な方は以下を実行してください。
（ファイル数が多いので時間がかかります。）

※　音声コーパスのインストールで、方法３（Ver1.0 のメディアからコピー）を実施された場合のみ実行してください。

```
$ ./sj2euc-ext.sh  /home/ximera/Ximera110/xdata/F009/corpus/16k
```

以上でインストールは完了です。
茶筌・南瓜の設定を行わないと XIMERA は正常に動作しません。
設定方法については、10 章「茶筌・南瓜設定」をご参照ください。

9．Java のインストール

以下のプログラムを実行するには Java が必要です。

- サンプルプログラム（XIMERA GUI）
- データベース作成 GUI （dbmake-gui）
- HTS 学習 GUI （hts-training）
- 音声切り出しツール（jspseg）
- F0・ラベル修正ツール（piikun）

これらのプログラムをコンパイル、実行するには、J2SE™ v1.4.2_04 を以下のサイトから入手し、インストールしてください。
http://java.sun.com/j2se/1.4.2/ja/download.html

また、Windows、Linux とも、Java インストールディレクトリ/bin にパスが通っていることを確認してください。

１０．茶筌・南瓜設定

茶筌の設定ファイル（chasenrc)、南瓜設定ファイル(cabocharc)をインストールした環境にあわせて編集します。絶対パスを指定する必要があるため、必ず行ってください。
また、Linux 版のみ、XIMERA GUI を動作させるためのパスを設定します。

１０．１ 茶筌・南瓜設定（Windows 版、Linux 版共通）

下記ファイルを編集します。

Windows の場合、

Ximera110¥xdata¥settings¥windows¥chasenrc

Ximera110¥xdata¥settings¥windows¥cabocharc

Linux の場合、

Ximera110/xdata/settings/linux/chasenrc

Ximera110/xdata/settings/ linux/cabocharc

chasenrc の編集サンプル

```
;;  chasenrc for ipadic-2.6.2
;;
;; 日本語ラベルと英語ラベルどちらか指定するだけで良い
;;;
;;;  grammar.cha/ctypes.cha/cforms.cha location /文法ファイル
;;;
;;;(GRAMMAR  ../../xdata/settings/windows/ipadic)
(GRAMMAR  c:/Ximera110/xdata/settings/windows/ipadic)

 (以降省略)
```

※ 網掛け部分にインストールしたディレクトリの絶対パスを記述してください。

※ ディレクトリ区切り文字には”¥”でなく”/”をご使用ください。

cabocharc の編集サンプル

```
(途中省略)

# Parser model file name
# parser-model  = ../../../xdata/settings/windows/cabocha/model/IPA-dep.model
parser-model  = c:/ximera110/xdata/settings/windows/cabocha/model/IPA-dep-A+P.model
# parser-model  = ../../../xdata/settings/windows/cabocha/model/IPA-small-dep.model
# parser-model  = ../../../xdata/settings/windows/cabocha/model/IPA-small-dep-A+P.model

# Chunker model file name
chunker-model = c:/ximera110/xdata/settings/windows/cabocha/model/IPA-chunker.model
# chunker-model = ../../../xdata/settings/windows/cabocha/model/JUMAN-chunker.model

# NE model file name
ne-model = c:/ximera110/xdata/settings/windows/cabocha/model/CRL-IREX-ne.model

# chasenrc (libchasen must be enabled.)
chasenrc = c:/ximera110/xdata/settings/windows/chasenrc
```

※ 網掛け部分にインストールしたディレクトリの絶対パスを記述してください。

※ ディレクトリ区切り文字には”¥”でなく”/”をご使用ください。

１０．２ XIMERA GUI 用設定（Linux 版のみ）

XIMERA GUI を動作させるため、環境変数 LD_LIBRARY_PATH に、libXimeraForJNI.so へのパスを追加します。

```
> export  LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/ximera/Ximera110/bin/linux
```

※ 上記例は、bash でライブラリーパスを追加する例です。

※ 網掛け部分はインストールディレクトリを記述してください。

※ 環境変数 LD_LIBRARY_PATH に、XIMERA Version1.1 以前のインストールディレクトリパスが設定されている場合は、設定を置き換えてください。

※ 環境変数設定ファイル等に記述いただくと便利です。

１１．HTK-align、HTS-1.1.1 の Make（HTK のダウンロードとパッチの当て方）

データベース作成ツールと HTS 学習ツールは The Hidden Markov Model Toolkit（以下 HTK)に含まれるいくつかのコマンドを利用しています。

XIMERA Version1.1 の配布メディアには、HTKのソース、HTKのソースから作成する実行体を含んでいません。データベース作成、HTS学習を行う場合は必ず次の作業を行ってください。

データベース作成用、HTS 学習用、それぞれについて HTK 3.2.1 をダウンロードし、パッチを当て、コンパイルを行ってください。

①データベース作成用

データベース作成に必要な以下の実行体を作成します。

HCopy, HParse, HVite

②HTS 学習用

HTS 学習に必要は以下の実行体を作成します。

HcompV, HERest, HHEd, HInit, HRest

Windows、Linux 別に手順を説明します。

１１．１ Windows の場合

Windows でパッチを当ててビルドする方法は以下のとおりです。

(1) http://htk.eng.cam.ac.uk/から HTK3.2.1.tar.gz をダウンロードし、解凍します。
※ 解凍に WinZip を使用した場合、正常にパッチがあたることを確認しております。
解凍ソフトによっては正常にパッチがあたらない場合もあります。ご注意ください。

(2) http://hp.vector.co.jp/authors/VA010446/toolbox1/ から patch254w.zip をダウンロード、解凍し、patch.exe を入手します。

【データベース作成用】

(1-1) Ximera110¥src¥htk-align に、解凍した HTK (ディレクトリ含)、patch.exe をコピーします。

(1-2) パッチを当てます。

DOS 画面で、Ximera110¥src¥htk-align ディレクトリに移動し、パッチを当てます。

```
> cd Ximera110¥src¥htk-align
> patch -p1 < win-htk-align.patch
```

(1-3) コンパイル

Microsoft Visual Studio.NET で Ximera110¥src¥htk-align¥htk.sln
を立ち上げビルド-バッチビルドメニューより、ビルドを実行してください。

(1-4) 実行環境へのコピー

Ximera110¥src¥htk-align¥install.bat を実行し、実行環境にコピーします。
HCopy.exe、HParse.exe、HVite.exe が、Ximera110¥src¥dbtool¥dbmake-gui¥winbin にコピーされます。

【HTS 学習用】

(2-1) Ximera110¥src¥hts-training¥HTS-1.1.1 に、解凍した HTK（ディレクトリ含）と patch.exe をコピーします。

(2-2) HTS のパッチを当てます

DOS 画面で、Ximera110¥src¥hts-training¥HTS-1.1.1 ディレクトリに移動し、パッチを当てます。

このパッチは、HTS から配布されているものと同じです。(http://hts.ics.nitech.ac.jp/)

```
> cd Ximera110¥src¥hts-training¥HTS-1.1.1
> patch -p1 < win-HTS-1.1.1_for_HTK-3.2.1.patch
```

(2-3) ATR のパッチを当てます

DOS 画面で、次のコマンドを実行します。

```
> patch -p1 < win-htk-train.patch
```

(2-4) コンパイル

Microsoft Visual Studio.NET で

Ximera110¥src¥hts-training¥HTS-1.1.1¥hts.sln

を立ち上げビルド-バッチビルドメニューより、ビルドを実行してください。

(2-5) 実行環境へのコピー

Ximera110¥src¥ hts-training¥HTS-1.1.1¥install.bat を実行し、実行環境にコピーします。

HcompV.exe, HERest.exe, HHEd.exe, HInit.exe, HRest.exe が、

Ximera110¥src¥dbtool¥training-gui¥winbin にコピーされます。

１１．２ Linux の場合

Linux でパッチを当ててビルドする方法は以下のとおりです。

(1) http://htk.eng.cam.ac.uk/から HTK3.2.1.tar.gz をダウンロードし、解凍します。

```
[ximera]> tar zxvf HTK-3.2.1.tar.gz
```

【データベース作成用】

(1-1) Ximera110/src/htk-align に解凍したソースをコピーします。

```
[ximera]> cp -rp htk/* /home/ximera/Ximera110/src/htk-align
```

(1-2) パッチを当てます。

Ximera110/src/htk-align　ディレクトリに移動し、パッチを当てます。

```
[ximera]> cd Ximera110/src/htk-align
[htk-align]> patch -p1 < linux-htk-align.patch
```

(1-3) Make の実行

```
[htk-align]> make
```

(1-4) 実行環境へのコピー

データベース作成の実行環境に作成した実行体をコピーします。

```
[htk-align]> make install
```

HCopy, HParse, HVite が、Ximera110/src/dbtool/dbmake-gui/linuxbin にコピーされます。

【HTS 学習用】

(2-1) Ximera110/src/hts-training/HTS-1.1.1 に解凍したソースをコピーします。

```
[ximera]> cp -rp htk/* /home/ximera/Ximera110/src/ hts-training/HTS-1.1.1
```

(2-2) HTS のパッチを当てます。

Ximera110/src/hts-training/HTS-1.1.1　ディレクトリに移動し、パッチを当てます。

このパッチは、HTS から配布されているものと同じです。(http://hts.ics.nitech.ac.jp/)

```
[ximera]> cd Ximera110/src/hts-training/HTS-1.1.1
[HTS-1.1.1]> patch -p1 < linux-HTS-1.1.1_for_HTK-3.2.1.patch
```

(2-3) ATR のパッチを当てます。

```
[HTS-1.1.1]> patch -p1 < linux-htk-train.patch
```

(2-4) Make の実行

```
[HTS-1.1.1]> make
```

(2-5) 実行環境へのコピー

データベース作成の実行環境に作成した実行体をコピーします。

```
[HTS-1.1.1]> make install
```

HcompV, HERest, HHEd, HInit, HRest が、Ximera110/src/dbtool/training-gui/linuxbin にコピーされます。

１２．ディレクトリ構成

XIMERA Version1.1のディレクトリ構成は以下のとおりです。

```
Ximera110
├──setupbin                インストール時に使用するシェル（Linuxのみ）
├──bin                     実行用のバイナリおよび、スクリプト
|   ├──linux               実行用のバイナリおよび、スクリプト Linux 用
|   └──windows             実行用のバイナリおよび、スクリプト Windows 用
├──doc                     ドキュメント
├──sample                  サンプル
|   ├──piikun              F0・ラベル修正ツールサンプルデータ
|   ├──sapixml             SAPIサンプルXMLファイル
|   └──ximera-gui          XIMERA GUI 用サンプルデータ
├──src                     ソースファイル
|   ├──bancha              数詞句解析ライブラリ
|   |       └──banlib      数詞句解析ライブラリソース
|   ├──bound               ポーズ挿入ライブラリ
|   ├──cabocha             係り受け解析ライブラリ
|   |       └──src         係り受け解析ライブラリソース
|   ├──chasen              茶筌ライブラリ
|   |       ├──darts       Double-ARray Trie System
|   |       ├──lib         茶筌ライブラリソース
|   |       └──mkchadic    辞書作成
|   ├──common              共通ライブラリソース
|   ├──concat              素片選択・接続ソース
|   ├──dbmake              データベース作成関連ソース
|   |       ├──add_htk_header  特徴データファイルにHTKヘッダを追加するソース
|   |       ├──bin_units   バイナリファイル作成ソース
|   |       ├──cont_vqinfo コードブックマージソース
|   |       ├──libdbmake   ライブラリソース
|   |       ├──mk_ccost    接続コストファイル作成ソース
|   |       ├──mk_logf0    F0データ変換ソース
|   |       ├──mk_phdtbl   音素環境代替コストテーブル作成ソース
|   |       ├──mk_units    unitsファイル作成ソース
|   |       ├──pitch       F0抽出ソース
```

```
|       ├─plinfo          言語韻律情報ファイル作成
|       |   ├─plinfo      言語韻律情報ファイル作成ソース
|       |   └─common      共通ライブラリソース
|       ├─pow             パワー抽出ソース
|       ├─resampf0        F0 リサンプリングソース
|       ├─select          セグメント選択ソース
|       ├─srconv          サンプリング周波数変換
|       |   ├─lpf         変換フィルターソース
|       |   ├─srconv      サンプリング周波数変換ソース
|       |   └─wavio       音声ファイル入出力ソース
|       ├─vqinfo          VQ 関連のソースのディレクトリ
|       |   ├─ext_frm     Unit Boundadry 周辺特徴抽出ソース
|       |   ├─include     インクルードファイル
|       |   ├─mk_cent     Generation of Phonetic Centroid ソース
|       |   ├─mk_phcov    分散共分散行列対角成分ファイル作成ソース
|       |   └─mk_vqif     VQ Information Vector 抽出ソース
|       ├─vq-kmean        VQ(K-mean アルゴリズム)計算ソース
|       └─vq-lbg          VQ(LBG アルゴリズム)計算ソース
|   ├─dbtool              データベース作成関連ツールソース
|       ├─dbmake-gui      データベース作成 GUI (Java)
|       ├─jspseg          音声切り出しツール(Java)
|       ├─piikun          F0・ラベル修正ツール(Java)
|       ├─pli2kana        カナファイル作成ツール
|       ├─training-gui    HTS 学習 GUI (Java)
|       └─txt2pli         言語韻律情報作成ツール (jspseg 用)
|   ├─htk-align           データベースラベリング (パッチ提供)
|   ├─hts                 HTS ソース
|   ├─hts-training        HTS 学習関連ソース
|       ├─HTS-1.1.1       HTS ソース (パッチ提供)
|       └─SPTK            STPK ソース
|   ├─mor2pli             形態素→言語韻律情報変換ライブラリ
|   ├─plinfo              言語韻律情報→拡張環境依存ラベル変換ライブラリ
|   ├─sample              サンプルプログラム
|       ├─xibatsu         サンプルプログラム
|       └─ximera-gui      XIMERA GUI ソース(Java)
|   ├─segfreq             頻度情報ファイル作成ソース
```

```
|   ├─straight                STRAIGHT
|   |     ├─straight-analysis  メルケプストラムの抽出ソース
|   |     ├─straight-synthesis 音声変換合成ソース (XIMERA GUI 用)
|   |     ├─include            インクルードファイル
|   |     ├─straight-sp        ストレートライブラリファイル
|   |     └─straight-sub       ストレートライブラリファイル
|   ├─ximera-api              XIMERA アプリケーションインターフェース
|   ├─ximera-sapi             SAPI 対応ソース
|   |     ├─SetupXimera        SAPI セットアッププログラムソース
|   |     └─Ximera             SAPI DLL ソース
|   ├─ximera-wrap             XIMERA ラッププログラム
|   |     └─ximera-jni         JNI ソース (XIMERA GUI で使用)
|   └─yamcha                  YamCha ソース
|
└xdata                        データベース
    ├─F009                    女性コーパス (詳細は「コーパス情報」をご参照ください)
    |   ├─corpus              コーパス(XIMERA Ver1.0 よりコピーしてください)
    |   |   ├─16k             16kHz 音声コーパス(XIMERA Ver1.0 よりコピーしてください)
    |   |   ├─48k             48kHz 音声コーパス(XIMERA Ver1.0 よりコピーしてください)
    |   |   └─hts16k          HTS 学習元データ(XIMERA Ver1.0 よりコピーしてください)
    |   ├─hts                 HTS 学習結果
    |   |   └─F009_hts        HTS 学習結果 F009_hts
    |   └─seg                 データベース
    |       ├─F009_503        データベース  F009_503
    |       └─F009            データベース  F009
    └─settings                辞書時関連設定ファイル
        ├─linux               Linux 用辞書関連設定ファイル
        |   ├─cabocha         南瓜モデルファイル
        |   |   └─model       南瓜モデルファイル
        |   └─ipadic          Linux 用辞書関連設定ファイル
        |       ├─srcdic      Linux 用辞書ファイル
        |       └─mkchadic    Linux 用辞書作成実行体
        └─windows             Windows 用辞書関連設定ファイル
            ├─cabocha         南瓜モデルファイル
            |   └─model       南瓜モデルファイル
            └─ipadic          Windows 用辞書関連設定ファイル
```

```
├─srcdic        Windows 用辞書ファイル
└─mkchadic      Windows 用辞書作成実行体
```

XIMERA（Ver1.0）

コーパス情報

２００４年３月３１日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

本書は、XIMERA Version1.0 で配布するコーパスについて説明します。
また、それらのコーパスから作成したデータベース、HMM について説明します。

1．話者について

すべての音声ファイルは、同一の女性ナレーターの声です。

2．コーパス内容

データベース用のコーパス（corpus/48k、corpus/16k 以下）と、
HTS 学習用コーパス（corpus/hts16k 以下）があります。

2．1 データベース用コーパス

データベース用コーパスとして配布するファイルは以下のとおりです。
音声ファイルは、48kHz 音声ファイルと、16kHz 音声ファイルを配布します。

Ximera100/xdata/F009/corpus/48k・16k の情報

コーパス名	内容	音声ファイル数	正味時間（時間）	48kﾃﾞｨﾚｸﾄﾘｻｲｽﾞ（GB）	48kHz音声 DVD-R番号	16kﾃﾞｨﾚｸﾄﾘｻｲｽﾞ（GB）	16kﾃﾞｨﾚｸﾄﾘ DVD-R番号
AOZORAR	小説	12979	17.27	8.12	2,3	2.88	3
ATR503	音素バランス文	503	0.58	0.29	1	0.11	1
BTEC	旅行会話	6940	3.63	2.51	1	0.92	1
CHIMEI	地名	4657	1.26	1.29	8	0.49	1
CHIMEI2	地名	6998	1.56	1.84	8	0.70	1
IPADIC	辞書	1000	0.22	0.26	1	0.10	1
MAINICHIR	毎日新聞	6189	9.77	4.40	4	1.55	2
MEI	名	6997	1.28	1.74	7	0.67	1
NIKKEIR	日経新聞	6161	10.69	4.70	5,6	1.65	2
SEI	姓	7000	1.34	1.76	7	0.67	1
WORD216	単語	216	0.05	0.06	1	0.02	1
VCHK	ボイスチェック	3721	2.52	1.53	9	0.55	1
合計		63361	50.15	28.49		10.31	

＜説明＞

(1) 16kHz 音声ファイルは、48kHz 音声ファイルを、サンプリングレート変換ツール srconv を使用して変換したものです。

(2) corpus/48k 以下のディレクトリには、サンプリングレート 48kHz の音声ファイルのみが存在します。データベースの設定ファイル data_wvc.txt を編集することで、48kHz での合成が可能となります。詳細は、インストールマニュアルをご参照ください。

(3) corpus/16k 以下のディレクトリには、下記ファイルが存在します。

コーパス	コーパス内容(*1)
ATR503	wav, kna, xf0, f0, mk
ATR503 以外	wav, kna, xf0

(*1)ファイル拡張子の説明

wav	音声ファイル
kna	カナファイル
xf0	F0 ファイル
f0	修正済み F0 ファイル
mk	修正済みラベルファイル

(4) F0 ファイル(*.xf0)は、STRAIGHT[1]のピッチ抽出モジュールを使用して求めたものです。ただし、STRAIGHTは著作権の問題で配布できないため、データベース作成GUIではPitch-Tracker[2]を使用しています。Pitch-Trackerで求めたF0 は、STRAIGHTで求めたものより、若干精度が劣ります。

(5) コーパス VCHK は、収録時の声質をチェックするために収録したコーパスです。配布していますデータベース F009、F009_503 作成には使用していません。

[1] STRAIGHT(Speech Transformation anf Representation based on Adaptive Interpolation of weiGHTed spectrogram)は、ATRで開発された音声合成分析ツールです。
[2] Pitch-Trackerは、ケンブリッジ大学で開発されたF0 抽出プログラムです。

2．2 HTS 学習用コーパス

HTS 学習用コーパスとして配布するファイルは以下のとおりです。

Ximera100/xdata/F009/corpus/hts16k の情報

コーパス名	音声ファイル数	コーパス内容 (*1)	コーパスサイズ (MB)
ATR503	503	wav, pli, mk. f0	105. 292
BTEC	665	wav, pli, xmk. xf0	94. 088
MAINICHIR	247	wav, pli, xmk. xf0	65. 276
NIKKEIR	251	wav, pli, xmk. xf0	69. 452
合計	1666		334. 108

(*1)ファイル拡張子の説明

pli　　言語韻律情報ファイル

xmk　　ラベルファイル

※ 他の拡張子はデータベース用音声コーパスをご参照ください。

＜説明＞

(1) HTS 学習用コーパスの、音声ファイル(*. wav)、ラベルファイル(*. xmk)、修正済みラベルファイル(*. mk)、F0 ファイル(*. xf0)、修正済 F0 ファイル(*. f0) はデータベース用音声コーパスと同じものです。

(2) F0 ファイル (*. xf0) は、データベース用コーパスと同様、STRAIGHT の F0 抽出モジュールを使用して求めたものです。

3．データベース

下記 2 つのデータベースを配布します。

Ximera100/xdata/F009/seg の情報

データベース名	含まれるコーパス	正味時間
F009_503	ATR503	0.5 時間
F009	AOZORAR, ATR503, BTEC, CHIMEI, CHIMEI2, IPADIC, MAINICHIR, MEI, NIKKEIR, SEI, WORD216 （VCHK 以外のすべてのコーパス）	47.6 時間

※ 正味時間とは、1 文単位で、発声前後の無音を省いた時間です。

4．HTS 学習結果

HTS 学習用コーパスより作成した HTS 学習結果を配布します。

Ximera100/xdata/F009/hts の情報

話者モデル名	学習に使用したコーパス	正味時間
F009_hts	corpus/hts16k 以下のすべて	1.79 時間

※ 正味時間とは、1 文単位で、発声前後の無音を省いた時間です。

XIMERA（Ver1.1）

プログラマーズマニュアル

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

本マニュアルは、ATR であらたに開発されたコーパスベース方式による音声波形素片接続型テキスト音声合成エンジン（開発コード名：XIMERA、以下「XIMERA」と記す）のコンパイル手順、XIMERA エンジン、XIMERA で使用するデータベース作成ツールと HTS 学習・HMM 作成ツール、Microsoft Speech API（以下「SAPI」と記す）対応アプリケーションから利用するためのプログラム（以下「XIMERA SAPI」と記す）について説明します。

1．コンパイル

1．1 Windows でのコンパイル

(1) エンジンのコンパイル

(1-1)プロジェクトファイルの編集

JNI用のヘッダーファイルが格納されているディレクトリへのパスをVisualStudio.NETに設定します。

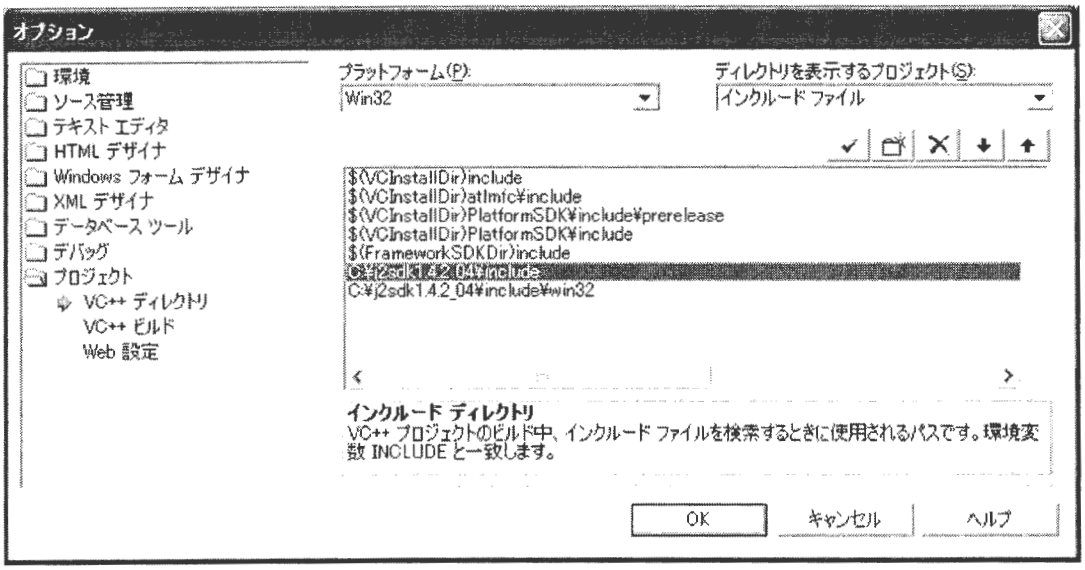
Ximera110¥src¥ximera_full.sln を開き、[ツール]→[オプション] で下記画面を開きます。

“プロジェクト”を選択し、画面右上のプロジェクトにインクルードファイルを選択し、下記インクルードディレクトリをリストに追加します。

- ＜ J2SE™ v1.4.2_04 InstallDir>¥include
- ＜ J2SE™ v1.4.2_04 InstallDir>¥include¥win32

※ コンパイルするにはJ2SE™ v1.4.2_04、Microsoft Speech SDK5.1 がインストールされている必要があります。

VisualStudio.NET オプション画面サンプル



(1-2)コンパイルの実行

[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。(バッチビルドウィンドウ中のすべてのプロジェクトのビルド欄がチェックされている事を

確認してください。)

(2) データベース作成ツールのコンパイル

Ximera110¥src¥dbtool¥pli.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、
Ximera110¥src¥dbtool¥install.bat を実行します。

(3) データベース作成 GUI 用　実行コマンドのコンパイル

Ximera110¥src¥dbmake¥dbmake.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、
Ximera110¥src¥dbmake¥install.bat を実行します。

(4) データベース作成用 align のコンパイル

HTK をダウンロードし、パッチがあたっていることが前提となります。
パッチの当て方については、インストールマニュアルをご参照ください。
Ximera110¥src¥htk-align¥htk.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、
Ximera110¥src¥htk-align¥install.bat を実行します。

(5) HTS のコンパイル

HTK をダウンロードし、パッチがあたっていることが前提となります。
パッチの当て方については、インストールマニュアルをご参照ください。
Ximera110¥src¥hts-training¥HTS-1.1.1¥hts.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、
Ximera110¥src¥hts-training¥HTS-1.1.1¥install.bat を実行します。

(6) SPTK のコンパイル

Ximera110¥src¥hts-training¥SPTK¥SPTK.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、
Ximera110¥src¥hts-training¥SPTK¥install.bat を実行します。

(7) STRAIGHT のコンパイル

Ximera110¥src¥straight¥straight.sln を開き、[ビルド]-[バッチビルド] メニューですべてのソースをコンパイルします。

作成した実行体を実行環境にコピーするには、Ximera110¥src¥straight¥install.bat を実行します。

(8) Java ソースプログラムのコンパイル

Java で記述されたソースをコンパイルします。
コンパイル対象は、データベース作成 GUI、HTS 学習 GUI、音声切り出しツール、F0・ラベル修正ツール、XIMERA GUI です。
Java がインストールされ、パスが通っていることが前提となります。
Ximera110¥src¥build-java.bat を実行します。

(9) XIMERA GUI 実行環境へのコピー

XIMERA GUI 実行環境を Ximera110¥bin 以下にコピーするには、
Ximera110¥src¥install.bat を実行します。

1．2 Linux でのコンパイル

(1) エンジンのコンパイル

(1-1)Make ファイルの編集

Ximera110/src/ximera-wrap/ximera-jni/Makefile に記述されている JNIDIR を JNI 用ヘッダーファイルが格納されているディレクトリに書き直します。

Ximera110/src/ximera-wrap/ximera-jni/Makefile

```
#   JNI library for ximera

#   edit this line to suit your environment
JNIDIR = /home/j2sdk1.4.2_04/include

LIBDIR = ./
 (省略)
```

※ 網掛け部分が編集箇所です。J2SE™インストールディレクトリを記述してください。

(1-2)Make の実行

Ximera110/src ディレクトリで Make を実行します。

```
[src]> make
```

(2) データベース作成、HTS 学習のコンパイル

HTK をダウンロードし、パッチがあたっていることが前提となります。
パッチの当て方については、インストールマニュアルをご参照ください。
Ximera110/src ディレクトリで Make を実行します。

```
[src]> make -f MakeDB
```

実行環境にコピーを行う場合、下記を実行します。

```
[src]> make -f MakeDB install
```

(3) Java コンパイル用シェルへの実行権限付与

(4)で使用する java コンパイル用シェルに実行権限を与えます。

シェル自体に実行権限を付与した後、Ximera110/src ディレクトリで実行してください。

```
[src]> chmod +x ../setupbin/chmode-sh.sh
[src]> ../setupbin/chmode-sh.sh
```

(4) Java ソースプログラムのコンパイル

Java で記述されたソースをコンパイルします。

コンパイル対象は、データベース作成 GUI、HTS 学習 GUI、音声切り出しツール、F0・ラベル修正ツール、XIMERA GUI です。

Java がインストールされ、パスが通っていることが前提となります。

```
[src]> ./make-java.sh
```

(5) 実行環境へのコピー

XIMERA GUI の実行に必要なファイルを bin 環境にコピーします。

```
[src]> ./install.sh
```

2．システム概要

本システム XIMERA は、仮名漢字混じりの日本語テキストより合成音声データを出力します。

本システムは、音声合成エンジン XIMERA、データベース、データベース作成に必要なツール、コーパスを提供しています。

コーパスとはデータベースに必要な元データ（音声ファイル、カナファイルなど）で、データベースとは、XIMERA エンジンでロードするファイル群です。

データベース作成ツールは、コーパスから自動で音声データの特徴抽出を行い、データベースの作成を行います。音声合成エンジンは、作成されたデータベース、コーパスを使用し、音声合成を行います。

音声合成エンジンは、音声合成する日本語テキストより読みアクセントの付与を行い、HTS を使用して韻律予測を行います。

音声合成エンジンは、大規模なコーパスから作成されるデータベースより、最適な音声素片を選択し、該当する音声データをコーパスから抽出します。エンジンは、抽出した音声素片データをつなぎ合わせ、合成音声データを作成します。

予測に近い素片を選択するため、素片の特徴をコスト値で表し、最適な素片をデータベースから選択します。このコスト値は、以下のコストを用いて計算しています。

F0 の違いに関するサブコスト（ SC_{F0} ）
音素持続時間の違いに関するサブコスト（ SC_{dur} ）
平均スペクトルの違いに関するサブコスト（ SC_{cep}）
音素環境代替に関するサブコスト（ SC_{env} ）
スペクトルの不連続に関するサブコスト（ SC_{spg} ）
F0 不連続性に関するサブコスト（ SC_{F0c} ）

※参考文献

戸田,河井,津崎,鹿野,“素片接続型日本語テキスト音声合成における音素単位とダイフォン単位に基づく素片選択”，電子情報通信学会論文誌，D-Ⅱ，Vol.J85-D-Ⅱ，No.12，pp.1760-1770，2002 年 12 月.

戸田智基，河井 恒，津崎 実，“波形接続型音声合成における知覚的評価に基づく素片選択サブコスト関数の最適化”，電子情報通信学会技術報告，SP2003-81，pp.43-48，2003

データベースは、上記コスト値を計算するために必要な特徴量をコーパス中の全音声素片に対して計算し、集積したものです。

3．機能一覧

以下は本システムでサポートする機能一覧を記述します。

音声合成エンジン関係の機能
・形態素解析（茶筌）
・辞書作成ツール
・辞書データ（ipadic にアクセント情報付加。固有名詞の追加）
・日本語係り受け解析
・数詞句解析
・同形語読み分け解析
・複合語内アクセント句境界解析
・複合語の連濁解析
・境界情報解析（ipadic の辞書を使用）
・アクセント句の分割および句のアクセント核位置の決定
・ポーズ挿入
・素片選択・接続

データベース関係の機能
・データベース作成ツール
・音声切り出しツール
・F0・ラベル修正ツール
・HTS 学習ツール

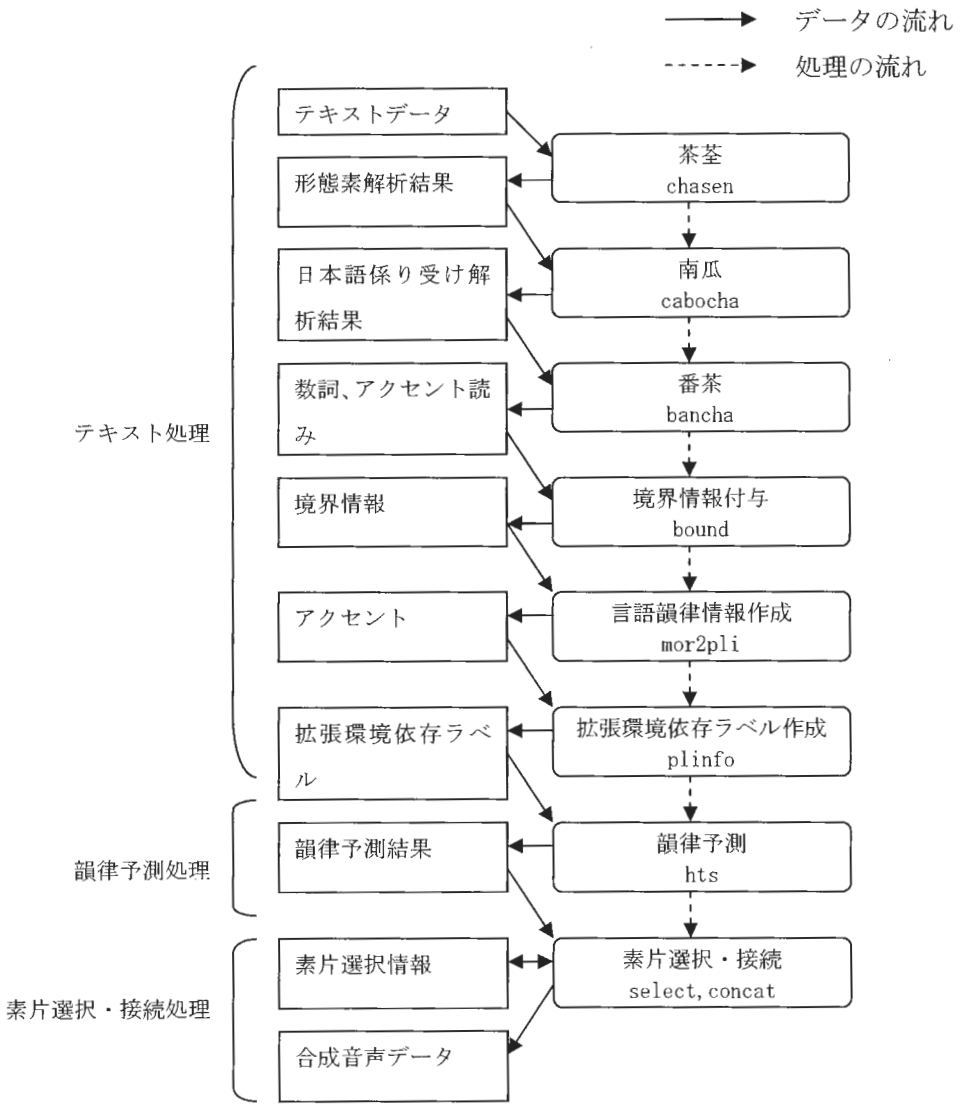
詳細について各モジュールにて説明します。

4．データフロー

音声合成エンジン、データベース作成ツールのデータの流れについて概要図を示します。

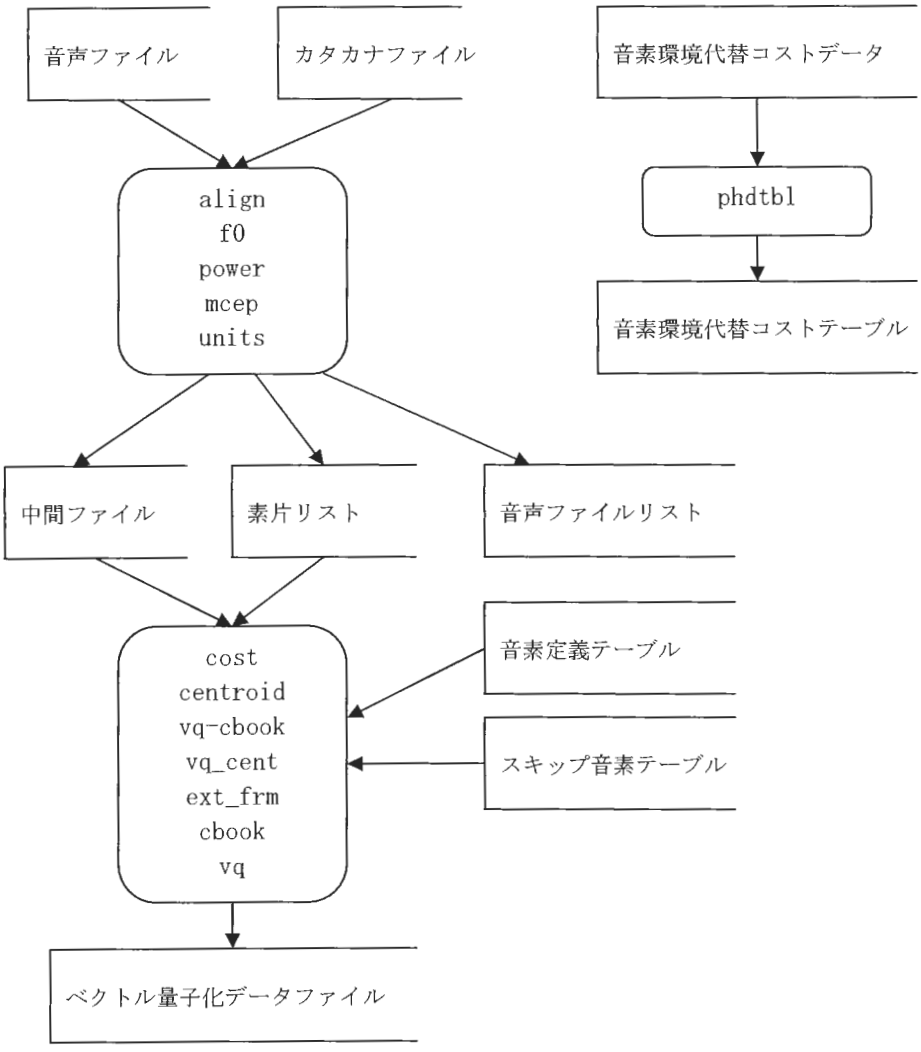
4．1 音声合成エンジン データフロー

音声合成エンジンは日本語かな混じりテキストデータを入力とし、下図の各モジュールで解析結果情報を付与し合成音声を作成します。



4．2 データベース作成 データフロー

データベース作成ツールは音声ファイル、カタカナファイルを入力とし、素片データの特徴量を抽出します。抽出した情報は、音声合成エンジンの素片選択・接続でデータベースとして使用されます。下図はデータベース作成時のデータの流れを示した概要図です。



詳細は７章を参照してください。

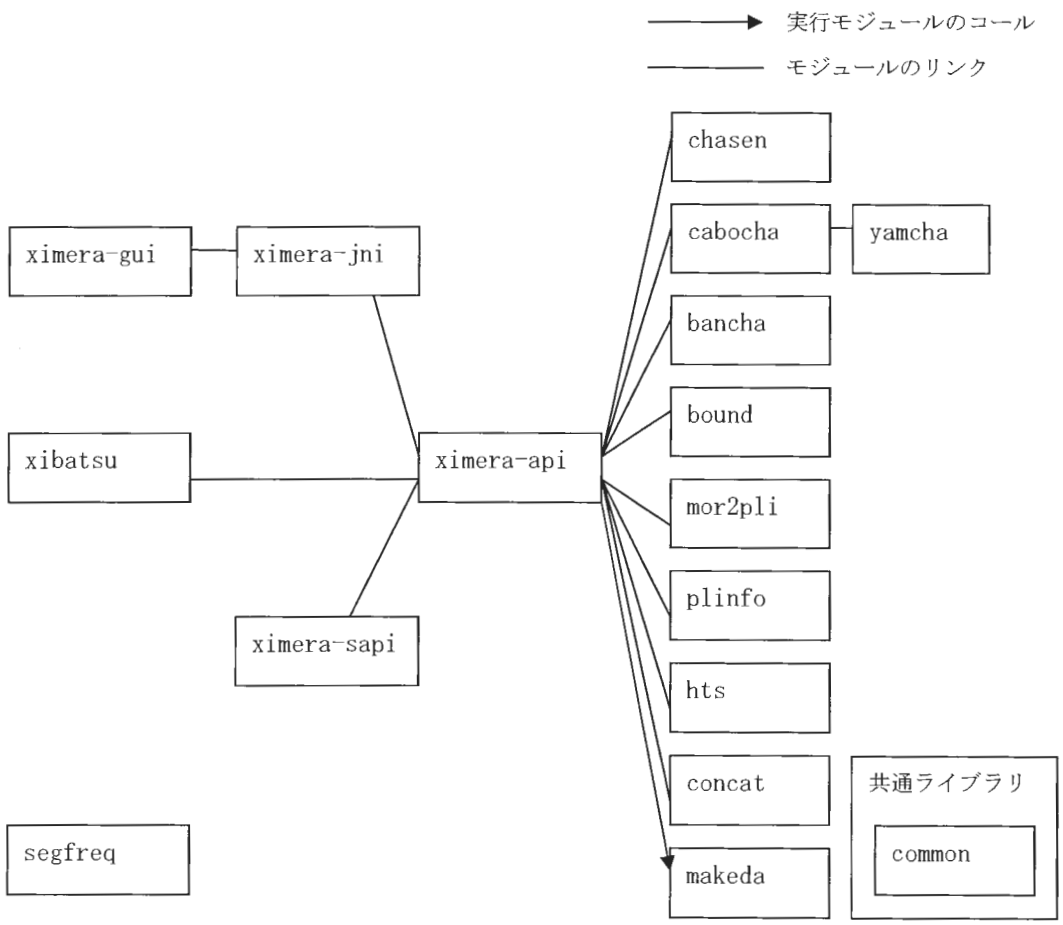
5．モジュール構成

XIMERA は、音声合成エンジン、サンプルアプリケーション、データベース作成ツール、HTS トレーニングツールから構成されています。

ここでは、各システムのモジュール構成を記述します。

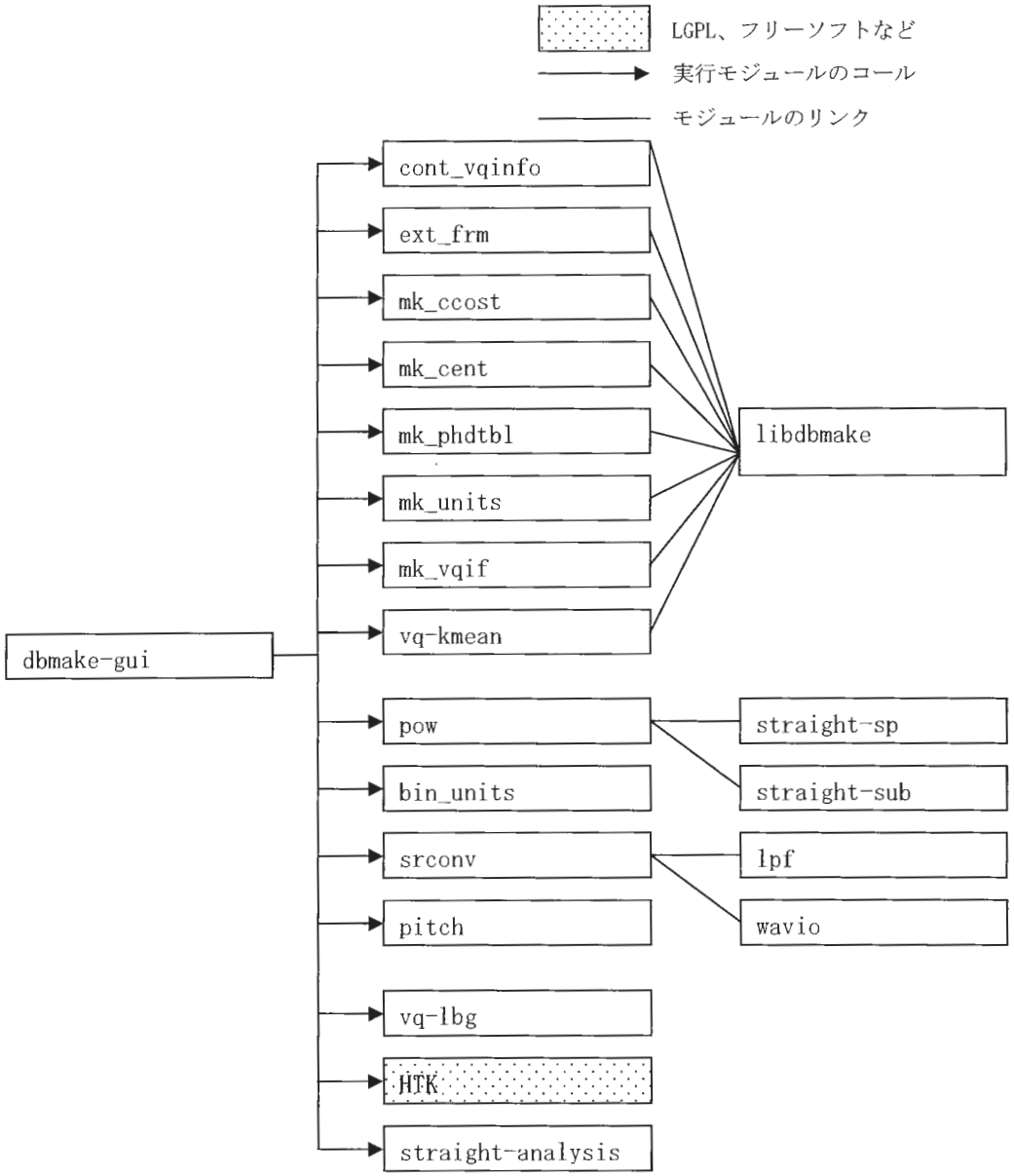
5．1 音声合成エンジンのモジュール構成

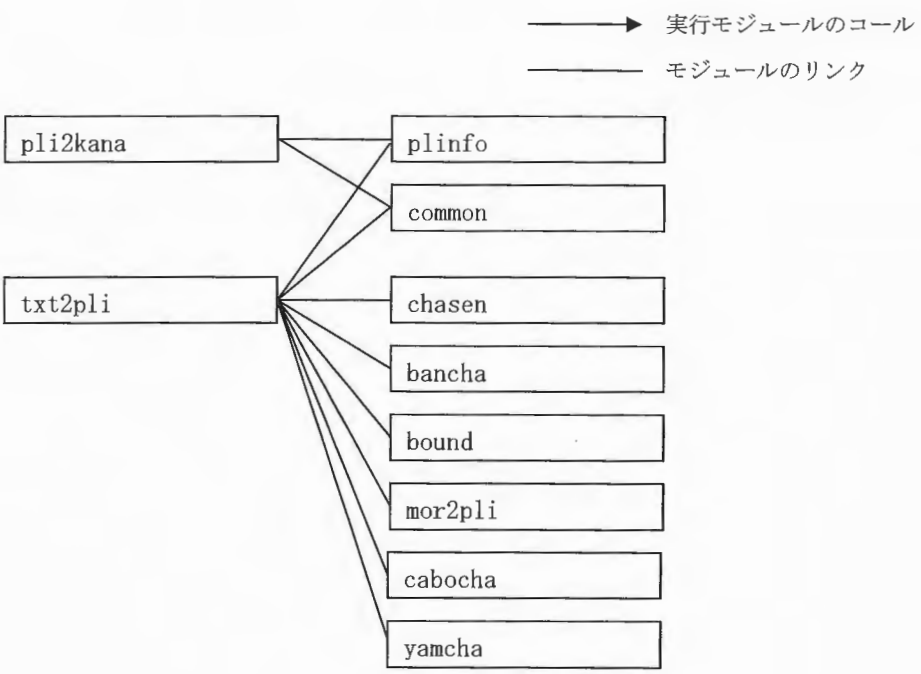
サンプルアプリケーションは XimeraAPI を通して音声合成エンジンを使用します。



5．2 データベース作成ツールのモジュール構成

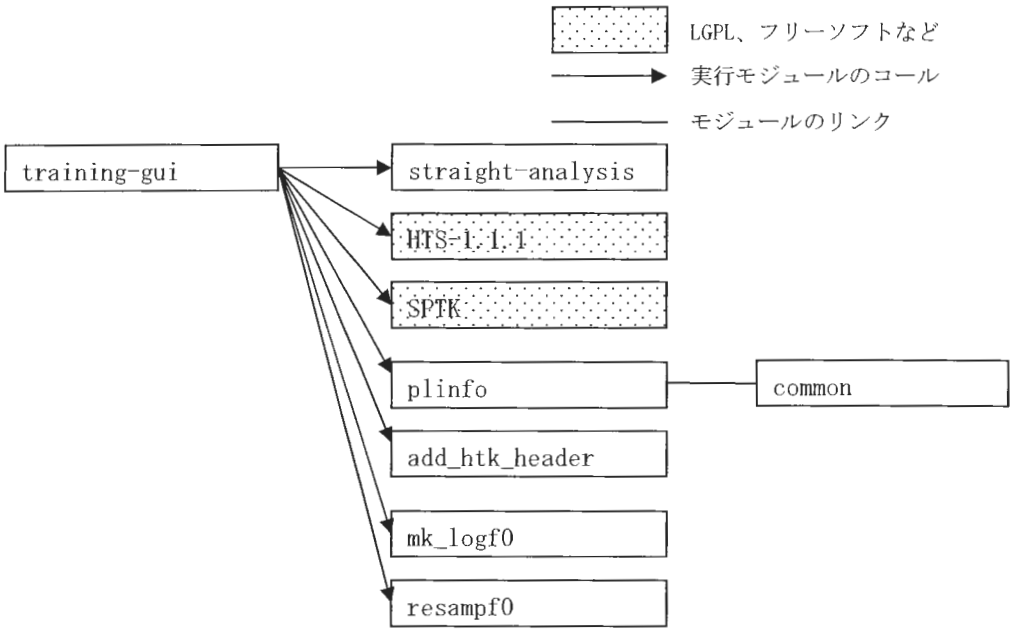
下図はデータベース作成ツールの GUI がデータベース作成の各機能をコールする関係図を示しています。





5．3 HTS トレーニングツールのモジュール構成

下図は HTS トレーニングツールの GUI が各機能をコールする関係図を示しています。



6．音声合成エンジン

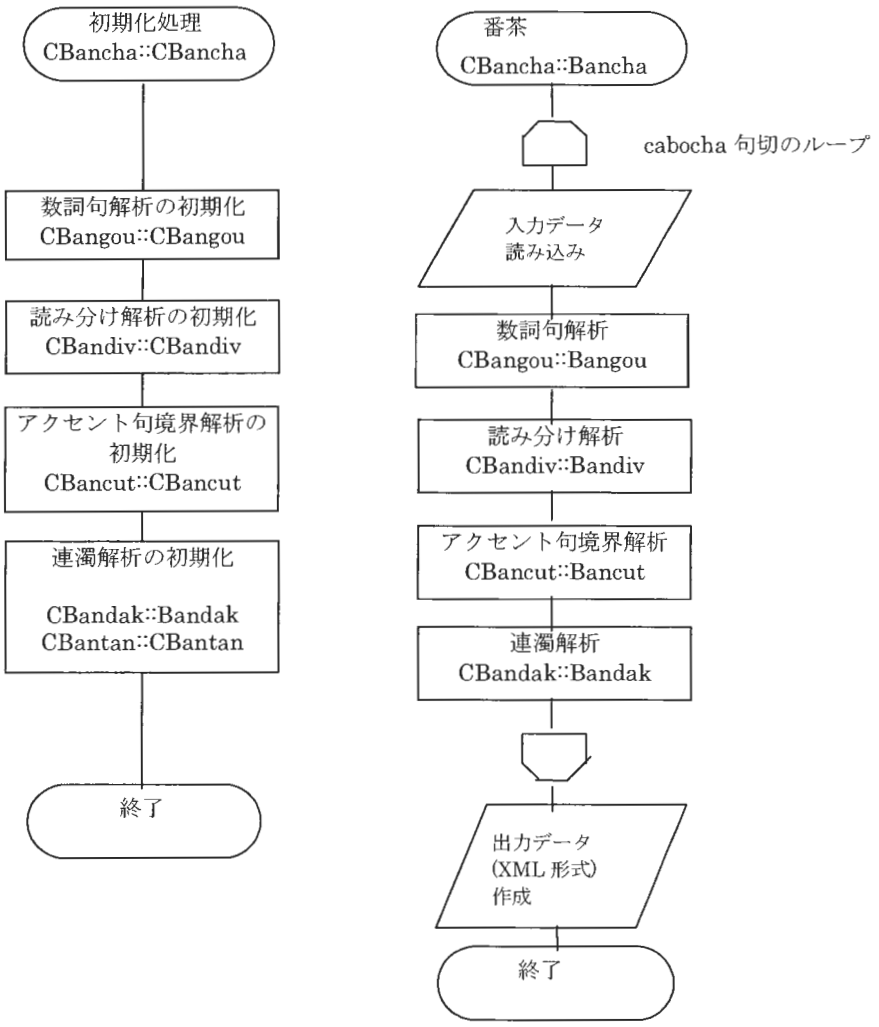
ここでは、音声合成エンジンの各モジュールの説明を記述します。

6．1 番茶

6．1．1 処理の流れ

番茶は形態素解析のポストプロセッサです。茶筌および南瓜の解析結果をもとに、「数詞句解析」「読み分け解析」「アクセント句境界解析」「連濁解析」を行います。

処理フロー



６．１．２ 入力フォーマット

番茶は茶筌の出力フォーマットのうち下記のフォーマットに対応しています。

chasenrc の設定

```
(OUTPUT_FORMAT "%m¥t%yO¥t%M¥t%U(%P-)¥t%T ¥t%F ¥t%aO¥t%iO¥n")
```

意味は次のとおりです。

テキスト [Tab] 読み [Tab] 基本形 [Tab] 品詞 [Tab] 活用型 [Tab] 活用形 [Tab] 発音 [Tab] 付加情報 [Tab] cabocha 固有表現

(*)付加情報には、アクセントおよびアクセント結合情報が設定されています。詳細は次章の「辞書の付加情報」をご覧ください。

(*)読み、発音、付加情報には「{…/…}」で区切ることにより複数とおりの情報を含めることができます。

例：「山本家は１４代続いた家です。」の茶筌および南瓜の出力結果。

```
* 0 3D 1/2 6.73575731
山本 ヤマモト 山本 名詞-固有名詞-人名-姓 ヤマモト accent=0-4:accent_con=* B-PERSON
家 {カ/ケ} 家 名詞-接尾-一般 {カ/ケ} {accent=1-1:accent_con=C4ge2elseC3/ (略)
は は ハ 助詞-係助詞 ワ accent=0-1:accent_con=名詞%F1,動詞%F2@0, (略)
* 1 2D 2/2 5.69013323
１ イチ 1 名詞-数 イチ accent=2-2:accent_con=C3 O
４ ヨン 4 名詞-数 ヨン accent=1-2:accent_con=C1 O
代 ダイ 代 名詞-接尾-助数詞 ダイ {accent=0-2:accent_con=C4/accent=1-2:accent_con=C4} O
* 2 3D 0/1 0.00000000
続い ツヅイ 続く 動詞-自立 五段・カ行イ音便 連用タ接続 ツズイ (略)
た た タ 助動詞 特殊・タ 基本形 タ {accent=1-1:accent_con=動詞%F1, (略)
* 3 -1O 0/1 0.00000000
家 {イエ/ウチ} 家 名詞-一般 {イエ/ウチ} {accent=2-2:accent_con=C4/accent=0-2:accent_con=C3} O
です デス です 助動詞 特殊・デス 基本形 デス {accent=1-2:accent_con=名詞%F2@1 , (略)
。 {。/クテン} 。 記号-句点 {。/クテン} {accent=*:accent_con=*/accent=0-3:accent_con=*} O
EOS
```

デフォルト動作（先頭のヨミを使用）では、
正しく読めない部分

6.1.3 出力フォーマット

```
<sentence>
    <chunk id=句番号
           link=係り先句番号（-1 なら係り先なし）
           rel=係りタイプ >
        <tok read=読み  base=基本形  pos=品詞
           ctype=活用型  cform=活用形 pron=発音 info=付加情報 >
           単語
        </tok>
    </chunk>
</sentence>
```

(D)係る、(A)同格、(P)並列
(O)係らない、(S)アクセント句境界

タグ	意味
sentence	文
chunk	句（南瓜の結果）
tok	形態素

例：入力フォーマットの例「山本家は１４代続いた家です。」に対する番茶処理の結果。

```
<sentence>
 <chunk id="0" link="3" rel="D">
  <tok read="ヤマモト" base="山本" pos="名詞-固有名詞-人名-姓"
      ctype="" cform="" pron="ヤマモト"
      info="{accent=0-4:accent_con=*/accent=0-4:accent_con=C2}" ne="B-PERSON">山本</tok>
  <tok read="ケ" base="家" pos="名詞-接尾-一般"
      ctype="" cform="" pron="ケ" info="accent=1-1:accent_con=C3" ne="O">家</tok>
  <tok read="ハ" base="は" pos="助詞-係助詞"
      ctype="" cform="" pron="ワ"
      info="accent=0-1:accent_con=名詞%F1,動詞%F2@0,形容詞%F2@0" ne="O">は</tok>
 </chunk>
 <chunk id="1" link="2" rel="D">
  <tok read="ジュウヨンダイ" base="１４代" pos="名詞-数" ctype="" cform=""
      pron="ジューヨンダイ" info="accent=0-2,2-4:accent_con=*" ne="0">１４代</tok>
 </chunk>
 <chunk id="2" link="3" rel="D">
  <tok read="ツヅイ" base="続く" pos="動詞-自立" ctype="五段・カ行イ音便"
      cform="連用タ接続" pron="ツズイ" info="accent=0-3:accent_con=*" ne="O">続い</tok>
  <tok read="タ" base="た" pos="助動詞" ctype="特殊・タ" cform="基本形" pron="タ"
      info="{accent=1-1:accent_con=動詞%F1,形容詞%F2@-2,特殊助動詞%F2@0/
              accent=1-1:accent_con=形容詞%F2@-1}" ne="O">た</tok>
 </chunk>
 <chunk id="3" link="-1" rel="O">
  <tok read="イエ" base="家" pos="名詞-一般" ctype="" cform=""
        pron="イエ" info="accent=2-2:accent_con=C4" ne="O">家</tok>
  <tok read="デス" base="です" pos="助動詞" ctype="特殊・デス" cform="基本形"
        pron="デス"    info="{accent=1-2:accent_con=名詞%F2@1,動詞%F1,形容詞%F2@0/
                    accent=1-2:accent_con=動詞%F2@0}" ne="O">です</tok>
  <tok read="。" base="。" pos="記号-句点" ctype="" cform="" pron="。" info="accent=*:accent_con=*"
ne="O">。</tok>
 </chunk>
</sentence>
```

読み分け解析の結果

数詞句解析の結果

6．1．4 数詞句解析

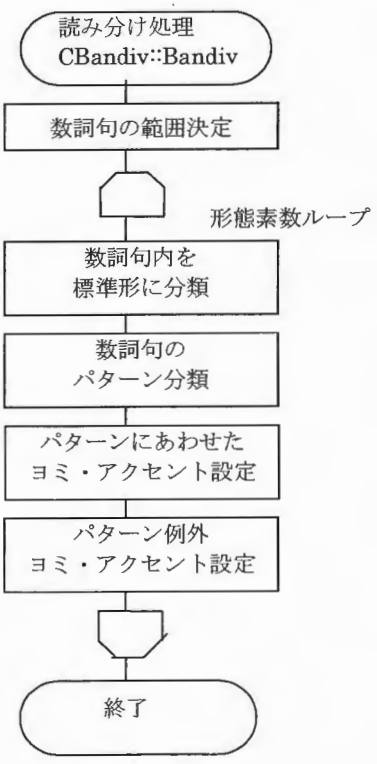
形態素解析結果のうち数詞の部分について、正しいヨミとアクセントを設定します。
例えば、「１２３４円」の形態素解析結果は以下のとおりです。

```
* 0 -1O 4/4 0.00000000
1　イチ 1　名詞-数　イチ　　accent=2-2:accent_con=C3　　B-MONEY
2　ニ　2　名詞-数　ニ　　accent=1-1:accent_con=C3　　I-MONEY
3　サン 3　名詞-数　サン　　accent=0-2:accent_con=C3　　I-MONEY
4　ヨン 4　名詞-数　ヨン　　accent=1-2:accent_con=C1　　I-MONEY
円　エン 円　名詞-接尾-助数詞　エン　　accent=1-2:accent_con=C3　　I-MONEY
EOS
```

数詞句解析を行い、「ｾﾞﾝ ﾆﾋｬｸ ｻﾞﾝｼﾞｭｳ ﾖﾞｴﾝ」と変換します。

```
<sentence>
 <chunk id="0" link="-1" rel="O">
  <tok read="センニヒャクサンジュウヨエン" base="１２３４円" pos="名詞-数"
        ctype="" cform="" pron="センニヒャクサンジューヨエン"
        info="accent=1-2,3-3,1-4,1-3:accent_con=*" ne="0">１２３４円</tok>
 </chunk>
</sentence>
```

数詞句解析処理フロー



６．１．４．１ 数詞句の範囲決定、項目分類

数詞句解析では、形態素解析結果から数詞句の形態素範囲を抽出、各項目に分類して処理を行います。

数詞句を以下のように定義します。

数詞句:=（前置助数詞）（符号）数詞（助数詞）（接辞）

各項目は以下のとおりです。

項目	内容
前置助数詞	・ 品詞が「接頭詞-数接続」のもの ・ 見出し語が bancharc の[前置助数詞]の定義にあるもの
符号	・ 見出し語が bancharc の[符号]の定義にあるもの
数詞	・ 見出し語が bancharc の[数字]の定義にあるもの
助数詞	・ 見出し語が bancharc の[助数詞]の定義にあるもの
接辞	・ 見出し語が bancharc の[接辞]の定義にあるもの

６．１．４．２ 数詞句のパターン分類（CBangou::Pattern）

数詞句がどのパターンかを判定します。

数詞句には「電話」「小数」「時間」「番地」などのパターンがあり、パターンごとに数詞の読み方を変えなくてはなりません。これらのパターンの正規形が bancharc の[パターン]に定義されています。

６．１．４．３ パターンにあわせたヨミ・アクセント設定（CNumRead::Reading）

パターンにあわせたヨミ・アクセントを設定します。アクセント変化は bancharc の定義のとおりです。

６．１．４．４ パターン例外のヨミ・アクセント設定（CNumRead::Reading）

以下の文字列はパターン適用の例外です。数詞句内の該当箇所に適用します。

見出し語	例外の結果
数日	{スウニチ/スウジツ}
一日 １日	{イチニチ/ツイタチ}
七日 ７日	{ナナニチ/シチニチ}
七人 ７人	{シチニン/ナナニン}
七年 ７年	{シチネン/ナナネン}
九日 ９日	{キュウニチ/クニチ}
九年 ９年	{クネン/キュウネン}

6.1.5 bancharc

数詞句解析の定義ファイルです。数詞句のアクセントなど定義します。定義は以下のフィールドに分かれています。行頭が";"の行はコメントとして扱います。

表記	内容説明
[前置助数詞]	前置助数詞とする語を定義します。
[符号]	符号とする語を定義します
[数字]	数詞の読みやアクセントなどの情報を定義します。
[助数詞]	助数詞とする語を定義します
[接辞]	接辞とする語を定義します
[パターン]	数詞句の種別を判定するパターンを定義します
[桁区切り]	桁区切りを判定するパターンを定義します
[アクセント]	数詞句の発音に対する読みとアクセントを定義します
[読み]	品詞が「名詞-数」の単語のうち、本機能での読み変更対象外とする語を定義します
[END]	終端

6.1.5.1 [前置助数詞]

フォーマットは以下のとおり。前置助数詞とする語を記述します。

```
単語
```

記述例

```
[前置助数詞]
〒
ＴＥＬ
番号
```

６．１．５．２［符号］

フォーマットは以下のとおり。符号とする語とその読みと発音を定義します。

```
単語　　Read=読み　Sound=発音
```

記述例

```
[符号]
±　Read=プラスマイナス　　Sound=プラスマイナス
プラス　Read=プラス　　Sound=プラス
```

６．１．５．３［数字］

フォーマットは以下のとおり。数詞の読みやアクセントなどの情報を定義します。

```
単語　Type=タイプ Read=読み Sound=発音 [Eng=英語読み] [Jap=漢数字]
単語　Tj=助数詞音韻変化
単語　Tk=アクセント変化
```

Type には、数字のタイプを定義します。以下の値が指定できます。

タイプ	意味
Jap	漢数詞（〇、一、二などに指定）
Eng	英数詞（０、１、２などに指定）
Unit	位取り（億、万などに指定）
Con	数字の一部として用いる語（ピリオドなどに指定）

Tj には、助数詞をこの単語に結合するときの音韻変化規則を定義します。値は複数個の数字並びです。以下の数値が指定できます。

数値	意味
0	音韻変化しない
1	濁音化する
2	半濁音化する

複数個の数字は[助数詞]の Tn 項目の２文字目に該当します。数字並びは左から、Tn 項目の２文字目のアルファベットに順に対応します（"A"が先頭）。

Tk には、助数詞をこの単語に結合するときのアクセント結合規則を定義します。値は区切り文字をはさんだ複数個の数字並びです。以下の数値が指定できます。

数値	意味
0	[助数詞]の各単語に指定されたアクセント結合規則（T1）に従う
1	平板
2	助数詞の第１モーラ
3	助数詞の最終モーラ

複数個の数字は[助数詞]の Tn 項目の 3 文字目に該当します。数字並びは左から、Tn 項目の 3 文字目のアルファベットに順に対応します（"A"が先頭)。

上記の数値を列挙するときの区切り文字には２種類あります。

区切り文字	意味
半角空白	数値の意味にしたがってアクセント結合する
!	直前の数字が２－９の場合は平板 それ以外ならば、数値の意味にしたがってアクセント結合する

記述例

```
[数字]
〇  Type=Jap     Read=レイ    Sound=レー
一  Type=Jap     Read=イチ    Sound=イチ
二  Type=Jap     Read=ニ Sound=ニ
三  Type=Jap     Read=サン    Sound=サン
（略）
; 英数字
0  Type=Eng     Eng=ゼロ     Jap=〇  Read=ゼロ    Sound=ゼロ
1  Tpe=Eng Eng=ワン     Jap=一  Read=ワン    Sound=ワン
2  Type=Eng     Eng=ツー     Jap=二  Read=ニ Sound=ニ
3  Type=Eng     Eng=スリー   Jap=三  Read=サン    Sound=サン
（略）
; 位取り
十  Type=Unit    Read=ジュウ      Sound=ジュー
百  Type=Unit    Read=ヒャク      Sound=ヒャク
千  Type=Unit    Read=セン        Sound=セン
万  Type=Unit    Read=マン        Sound=マン
億  Type=Unit    Read=オク        Sound=オク
兆  Type=Unit    Read=チョウ      Sound=チョー
（略）
; 数字の一部
,    Type=Con     Read=,
```

```
.    Type=Con     Read=テン    Sound=テン
～   Type=Con     Read=カラ    Sound=カラ
（略）
;Tj
;助数詞音韻変化
;        ABCDEFGHIJKLMNOPQR          [助数詞]Tn の２文字目に該当
○   Tj=000000000000000000
一   Tj=022000220002220002
二   Tj=000000000000000000
三   Tj=021111211202000002
（略）
;Tk
;アクセント変化
;                                    [助数詞]Tn の３文字目に該当
;        A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
○   Tk= 0 0 0 0 2 3 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0 0 0
一   Tk= 0 0 0 0 2 3 1 3 2 3 0 3 2 2 3 3 3 3 0 0 3 0 1 2 3 0
二   Tk= 0 0 0 0 2 3 1 0 2 3 0 3 2 2 3 0 0 0 0 0 0 0 0 2 3 3
三   Tk= 0 0 1 1 2 0 1 0 0 0 2 0 0 2 3 0 0 0 0 0 0 1 0 0 0 2
（略）
十   Tk= 0 0 0 0!2!3!1!3 0 0 0 3 2 2 3!3!3 0 0!3 0 0 0 0 0 0
百   Tk= 0 0 0 0!2!3!1!3 0 0 0 3 2 2 3!3!3 0 0!3 0 0 0 0 0 0
（略）
```

６．１．５．４［助数詞］

フォーマットは以下のとおり。助数詞とする語とその読みと発音を定義します。

```
単語   Read=読み Sound=発音 Tn=変化フラグ Tw=和語読み Tn   Tl=結合情報
```

Tn には、変化フラグとして英字３文字（例：Tn=BAA）を指定します。

Tw には、和語読みの場合の Tn を指定します。

英字	意味
１文字目	助数詞が接続した場合の数字の音韻変化（Ti）を指定します 後述の Ti の変化表を参照してください
２文字目	[数字]の Tj 項目の並びのどれに該当するかを指定します。 A が一番左の先頭数字にあたります。
３文字目	[数字]の Tk 項目の並びのどれに該当するかを指定します。 A が一番左の先頭数字にあたります。

Tl に指定する結合情報は、次章の「辞書の付加情報」の複合語アクセント結合規則(C1-C4)

を指定します。

Ti の音韻変化表です。縦軸が数詞、横軸が音韻変化のタイプです。

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
一							ｲｯ	ｲｯ		ｲｯ		ｲｯ	ｲｯ	ｲｯ	ｲｯ
二															
三															
四		ヨ	ヨ	ﾖｯ	シ									シ	
五															
六						ﾛｯ	ﾛｯ		ﾛｯ			ﾛｯ		ﾛｯ	
七			ｼﾁ	ｼﾁ	ｼﾁ								ｼﾁ		
八						ﾊｯ	ﾊｯ	ﾊｯ						ﾊｯ	ﾊｯ
九			ク	ク	ク										
十							ｼﾞｭｯ								
百						ﾋｬｯ	ﾋｬｯ		ﾋｬｯ			ﾋｬｯ			ﾋｬｯ

記述例

```
[助数詞]
円　　Read=エン　Sound=エン　Tn=BAG　Tw=Tl=C3
ヵ月　　Read=カゲツSound=カゲツ　Tn=LAA　Tw=　Tl=C1
ＧＢ　　Read=ギガバイト Sound=ギガバイト　Tn=AAA　　Tw=Tl=C1
```

６．１．５．５［接辞］

フォーマットは以下のとおり。接辞とする語を記述します。

```
単語
```

記述例

```
[接辞]
未満
以上
```

６．１．５．６［パターン］

フォーマットは以下のとおり。数詞句の種別を判定するパターンを定義します。

```
パターン名　前置助数詞;符号;数字;助数詞;接辞
```

前置助数詞や符号などは以下の文字を使用した正規表現で定義します。各項目が複数ある場合は"/"で複数指定します。

文字	意味
J	漢数詞 or 位取り
j	漢数詞
N	英数詞 or 位取り
n	英数詞
B	英数詞 or 漢数詞 or 位取り
b	英数字 or 漢数詞
u	位取り
@	何でもＯＫ，無ければＮＧ
0	無し
!	否定（指定文字以外）
’	限定（指定文字のみ）
+	この長さ以上
-	この長さ以下
#	長さ限定

記述例

```
[パターン]
小数=*;*;'○・J/'○．J/'○点 J/'０．N/'０・N/;*;*;
電話='電話番号/'電話/'ＴＥＬ/;0;+4b/+1b'・*/+1b'―*/;'番/0/;*;
時間=*;*;J/n/b'／b/;'年/'月/'日/'時/'分/'秒/'期/;*;
英語='Ｖ/'ワースト/'ナンバー/'ベスト/'トップ/;0;#1n/'１０/;0;*;
```

６．１．５．７［桁区切り］

桁区切りの判定のために利用するパターンです。桁区切りで分割された桁ごとに、数詞句の変換を行います。フォーマットは[パターン]と同じです。

6.1.5.8 [アクセント]

フォーマットは以下のとおり。数詞句の発音に対する読みとアクセントを定義します。

発音　Read=読み　Accent=アクセント　Flags=アクセント変化フラグ

Flags には、変化フラグとして数字５文字（例：Flags=30000）を指定します。

数字位置	意味
１文字目	後ろに"点"がつく場合のアクセント
２文字目	直後が１～９ならば平板にするフラグ（0：off，1:on）
３文字目	未使用
４文字目	0:デフォルト 1:「万」「億」「兆」に指定 2:「点」「秒」に指定 3:「カラ」に指定
５文字目	棒読み可能（0：off，1:on）

記述例

```
[アクセント]
ロッピャク　Read=ロッピャク　Accent=4-4　　Flag=30000
レー　　Read=レイ　　Accent=1-2　　Flag=10001
ニヒャク　Read=ニヒャク　Accent=3-3　　Flag=20000
```

6．1．5．9［読み］

フォーマットは以下のとおりです。品詞が「名詞-数」の単語のうち、本機能で読み変更を行わない語を記述します。

単語　Len=長さ（バイト数）　Read=読み　Sound=発音

記述例

```
[読み]
ゼロ    Len=4   Read=ゼロ      Sound=ゼロ
ひと    Len=4   Read=ヒト      Sound=ヒト
いく    Len=4   Read=イク      Sound=イク
ひゃく  Len=6   Read=ヒャク    Sound=ヒャク
```

６．１．６ 読み分け解析

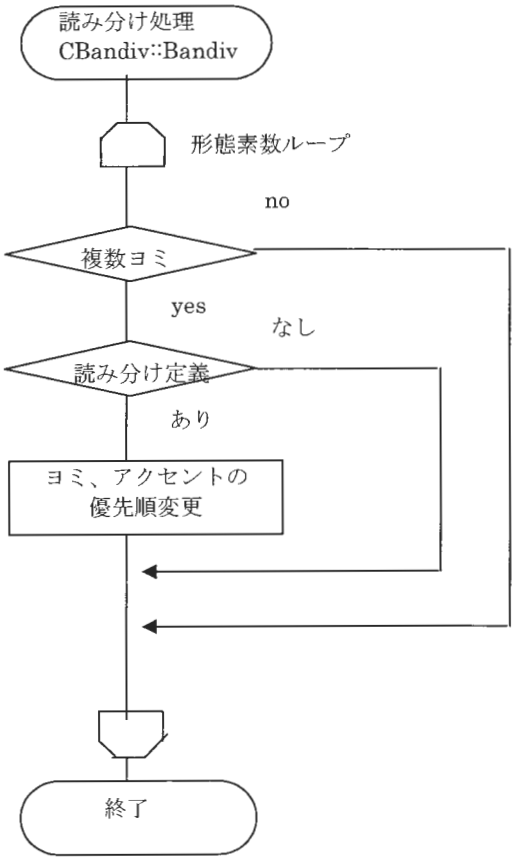
複数の読みを持つ語について、定義ファイル（bandivrc）にしたがって読みを一つに絞り込みます。デフォルトでは辞書の先頭のヨミを使用します。

例えば、「者」（名詞-接尾-一般）には、「シャ」「モノ」という２つのヨミがあります。

```
者　{シャ/モノ}　　者　名詞-接尾-一般 {シャ/モノ}　　{accent=1-1:accent_con=C3/（略）
```

「担当者」「前任者」などは優先度の高い「シャ」そのままで問題ありません。しかし、「働き者」のように「動詞」が前に来る場合は「モノ」と読み分けた方が適切です。読み分け解析は、このような読み分けパターンを定義したbandivrcにしたがって動作します。

読み分け解析処理フロー



6．1．7 bandivrc

読み分け解析の定義ファイルです。複数のヨミをもつ見出し語について、読み分けを行う条件を定義します。

項目	内容説明
見出し語 品詞 読み 読み分けする条件	読み分けする語 条件：前品詞、後品詞、前活用形（いずれか１つが指定）
[END]	終端

例

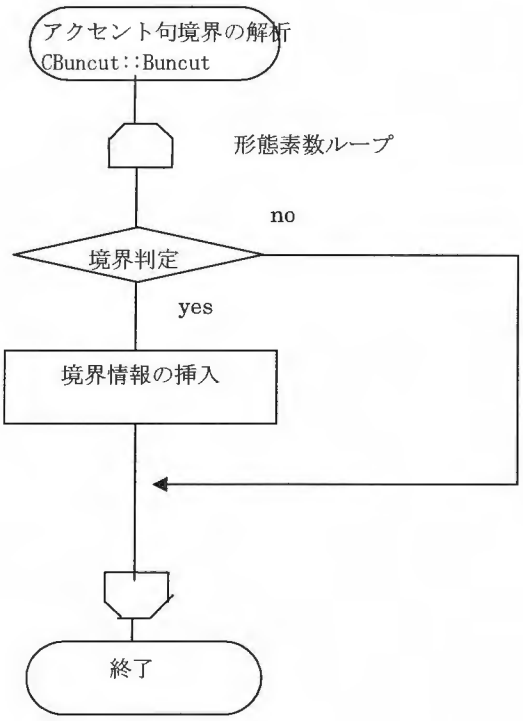
```
((見出し語 者)（品詞 名詞-接尾-一般）（読み モノ）（前品詞｛形容詞-自立/助詞-連体化/助動詞/接頭詞-名詞接続/動詞-自立/動詞-接尾/副詞-助詞類接続｝））
[END]
```

６．１．８ アクセント句境界の解析

連濁規則の適用は複合語に対して行うものです。連濁規則の適用範囲を限定するため、連濁解析の前段階としてアクセント句境界を解析します。

定義ファイル（bancutrc）に記述された境界判定規則にもとづき、アクセント句境界を付与します。判定に使用する条件は、品詞、活用形、活用型、結合情報です。

アクセント句境界の解析処理フロー



6.1.9 bancutrc

アクセント句境界解析について定義します。

[JUDGE]セクションで、品詞や活用語のまとまりで条件名を定義します。

[REQ]セクションでは、[JUDGE]で定義した条件について、境界挿入の動作を記述します。

表記	内容説明
[REQ]	条件群。
<Is ～ >	条件名　　エントリ全体を表現する固有名
キーワード　　値	キーワード：品詞、活用型、活用形、結合型、条件 値：アクセント結合型、条件名 各行はORで条件設定がされ、JUDGEから参照されます。
<End>	条件の終端
[END]	終端
[JUDGE]	アクセント句境界を判定する条件を記述します。
CUT　Is？？？　条件名 TIE　Is？？？　条件名	CUT アクセント句境界とする（分割） TIE　アクセント句境界としない（結合） Is？？？　条件名 (???)　比較する形態素 prev：境界前の形態素 line：境界後の形態素 next：次の形態素
[END]	終端

例：前品詞が接頭詞の時アクセント句境界とせず、本品詞が接頭辞の時は境界とします。

```
[REQ]
<Is 接頭詞>
品詞　　接頭詞-名詞接続
品詞　　接頭詞-動詞接続
品詞　　接頭詞-形容詞接続
品詞　　接頭詞-数接続
<End>
[END]

[JUDGE]
TIE Is接頭詞(prev)
CUT Is接頭詞(line)
[END]
```

６．１．１０ 連濁解析

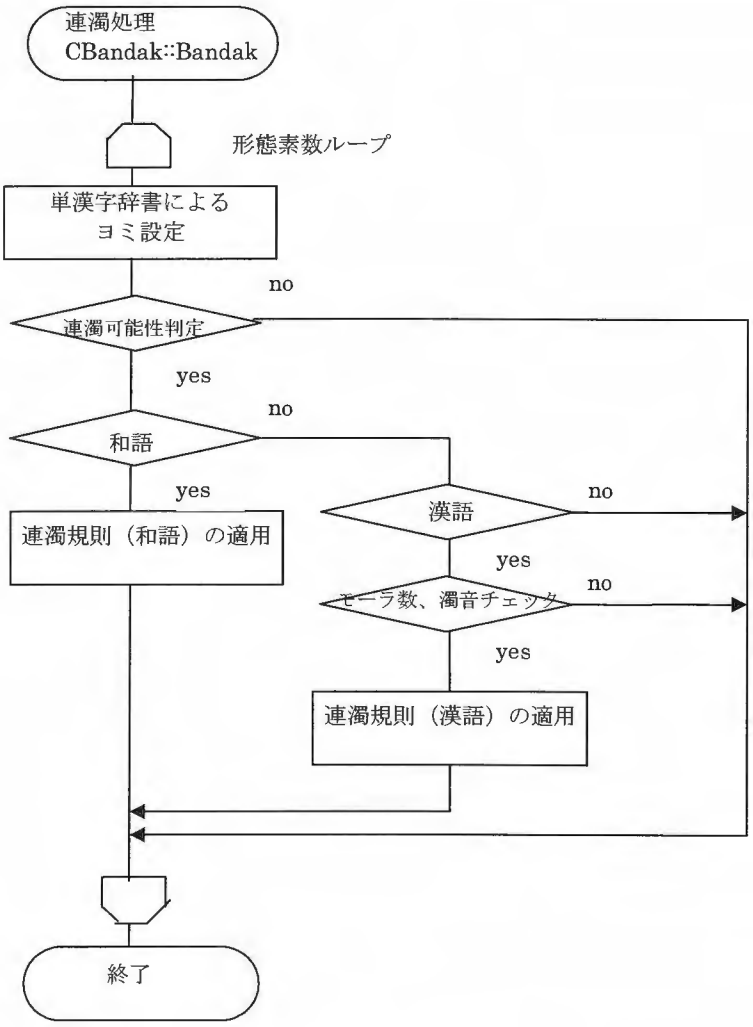
・単漢字辞書によるヨミ設定

ipadic を使用した形態素解析では、JIS 第二水準の漢字など難しい漢字が未知語になることがあります。ここでは、辞書に存在しない単漢字（人名、地名等の固有名詞、JIS 第二水準の漢字を含む語など）に読みを付与します。単漢字のヨミ定義は bancutrc です。

・連濁処理

連濁判定の前処理として、ヨミの濁音の有無や品詞などから連濁可能性を判定します。連濁の可能性がある場合、定義ファイル（bandakrc）にしたがって連濁処理します。

連濁解析処理フロー



６．１．１０．１ 連濁可能性の判定

形態素が以下のいずれかの条件にあてはまる場合、連濁規則の対象外とします。

項目	連濁を適用しない条件
位置	句の先頭
見出し語の先頭文字	漢字でない
読み	以下のいずれかを満たす場合 ・ 先頭カナが清音でない ・ 第２モーラ以降に濁音がある
品詞	前の形態素の品詞が以下のいずれかの場合 ・ 記号-* ・ 助詞-* ・ 助動詞 ・ 接続詞 ・ 連体詞 ・ 接頭辞-* 現在の形態素の品詞が以下のいずれかの場合 ・ 名詞-数 ・ 固有名詞-* 前と現在の形態素の品詞が以下の場合 ・動詞-* ＋ 動詞-*
見出し語の開始文字	以下のいずれかを満たす場合 ・ 「先」 ・ 「服」

６．１．１０．２ 連濁規則の適用

形態素の先頭の見出し語が「和語」か「漢語」かで処理が異なります。判定は、CBantan::IsType(見出し語,読み,CBantan::{Kun |On})でしています。

見出し語の先頭文字	連濁
和語	bandakrc に定義があれば連濁。
漢語	以下のいずれかを満たす場合は、連濁しない。 ・ 前の形態素の語が１モーラの場合 ・ 前の形態素の語が２モーラで、かつ、濁音がある場合 その他の場合、bandakrc に定義があれば連濁。

６．１．１１ bantanrc

単漢字のヨミ定義ファイルです。

以下のフォーマットで指定します。

表記	内容説明
単漢字:ID	同じ ID の単漢字は同じ文字とみなす
:（半角）	セパレータ
ID:タイプ読み	D0:ひらかな：訓読み（和語として扱う） 　カタカナ：音読み（漢語として扱う） E0:漢語読み F5:名前読み
:（半角）	終端

例

```
会:01190
會:01190
:
01190:D0 エ
01190:D0 カイ
01190:E0 ケ
01190:F5 あい
01190:F5 あう
01190:F5 もち
:
```

６．１．１２ bandakrc

連濁する可能性のある語について定義します。連濁規則は「和語」と「漢語」で異なるため、ここではそれぞれのセクションを分けて、対象の語と読みを列挙します。

以下のフォーマットで指定します。

表記	内容説明
[和語]	連濁する和語セクション
単語[TAB]読み	
[漢語]	連濁する漢語セクション
単語[TAB]読み	
[END]	終端

例

```
[和語]
火屋　　ヒヤ
澄む　　スム
下手　　ヘタ
勝ち　　カチ
[漢語]
会社　　カイシャ
合戦　　カッセン
菓子　　カシ
豆腐　　トウフ
布団　　フトン
砂糖　　サトウ
包丁　　ホウチョウ
[END]
```

６．１．１３ リファレンス

include

#include "banlib.h"

(1)コンストラクタ

CBancha::CBancha(const string& path);		
機能	初期化と、関連ファイルの読込み	
引数	const string& path	bancharc など関連ファイル群の 格納ディレクトリパス

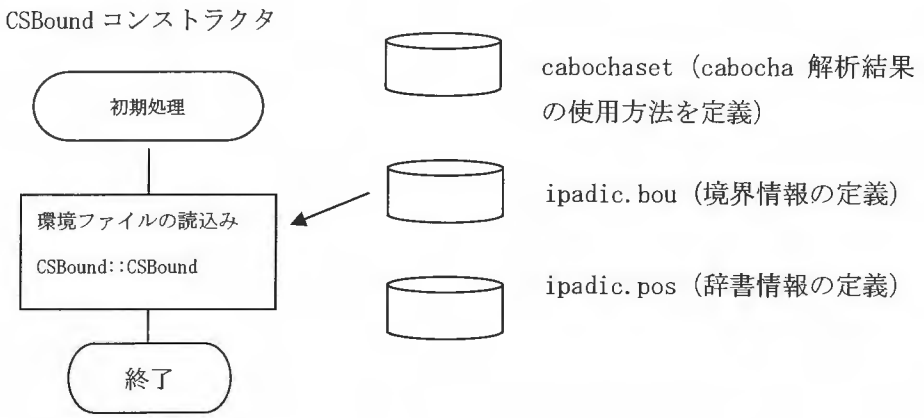
(2)解析処理

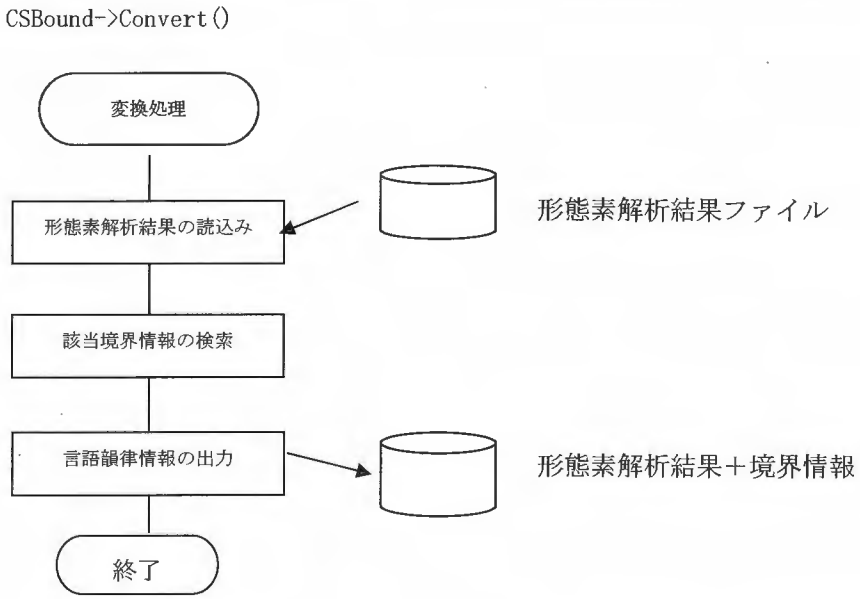
string CBancha::Bancha(const string& inStr);		
機能	解析	
引数	inStr	形態素解析データ（テキスト）
戻り値	string	解析結果(テキスト)

6．2 境界情報解析

本モジュール（CSBound クラス）は、境界情報の解析を行います。
前提辞書は、ipadic です。

全体の処理の流れ





6．2．1 入力フォーマット

番茶の出力フォーマットです。番茶の出力フォーマットを参照してください。

6.2.2 出力フォーマット

番茶の出力フォーマットに境界情報が付与されます。

```
<sentence>
    <chunk id=句番号
           link=係り先句番号（-1 なら係り先なし）
           rel=係りタイプ >
        <tok read=読み  base=基本形  pos=品詞
           ctype=活用型  cform=活用形 pron=発音 info=付加情報
           btype=境界情報 >
           単語
        </tok>
    </chunk>
</sentence>
```

境界情報解析で追加される属性

出力フォーマット例

```
<sentence>
 <chunk id="0" link="1" rel="D">
  <tok read="アラユル" base="あらゆる" pos="連体詞" ctype="" cform="" pron="アラユル" btype="係り受け" info="accent=3-4:accent_con=*" ne="0">あらゆる</tok>
 </chunk>
 <chunk id="1" link="2" rel="D">
  <tok read="ゲンジツ" base="現実" pos="名詞-一般" ctype="" cform="" pron="ゲンジツ" info="accent=0-4:accent_con=C2" ne="0">現実</tok>
  <tok read="ヲ" base="を" pos="助詞-格助詞-一般" ctype="" cform="" pron="オ" btype="係り受け" info="accent=0-1:accent_con=名詞%F1" ne="0">を</tok>
 </chunk>
 <chunk id="2" link="-1" rel="0">
  <tok read="ネジマゲ" base="ねじ曲げる" pos="動詞-自立" ctype="一段" cform="連用形" pron="ネジマゲ" info="{accent=4-5:accent_con=*/accent=0-5:accent_con=*}" ne="0">ねじ曲げ</tok>
  <tok read="タ" base="た" pos="助動詞" ctype="特殊・タ" cform="基本形" pron="タ" info="accent=1-1:accent_con=動詞%F1,形容詞%F2@-1" ne="0">た</tok>
 </chunk>
 <chunk id="2A" link="-1" rel="S">
  <tok read="ノ" base="の" pos="名詞-非自立-一般" ctype="" cform="" pron="ノ" info="accent=0-1:accent_con=動詞%F2@0" ne="0">の</tok>
  <tok read="ダ" base="だ" pos="助動詞" ctype="特殊・ダ" cform="基本形" pron="ダ" info="accent=1-1:accent_con=名詞%F1,動詞%F1" ne="0">だ</tok>
  <tok read="。" base="。" pos="記号-句点" ctype="" cform="" pron="。" info="accent=*:accent_con=*" ne="0">。</tok>
 </chunk>
</sentence>
```

6．2．3 cabochaset

用途：CaboCha 使用の有無、および、CaboCha 解析結果の使用方法の定義を行います。

文頭が"#"の行はコメント。
"[]"で囲まれた部分が項目名であり、空白で区切った後の文字列が値です。

```
#
# cabocha 関連機能の定義
#
[UseCabocha]    false

[BoundarySet]
  係り受け          linkplus=1
  非修飾境界３      linkplus=3..*  lpos=記号-読点 lpos=記号-句点
  非修飾境界３      linkplus=3..*  mcle=10..*
  非修飾境界３      linkplus=2     lpos=記号-読点 lpos=記号-句点
  非修飾境界２      linkplus=3..*
  非修飾境界２      linkplus=2     mci=10..*
```

UseCabocha

XIMERA で CaboCha を使用したテキスト解析を行うか否かを、"true/false"で指定します。
"true"の場合、CaboCha の解析結果を元に、ポーズ挿入に必要な境界情報を決定します。

BoundarySet

CaboCha の解析結果と境界情報を定義します。([UseCabocha] true の場合有効)
行先頭には境界種別を指定します。それ以降は、以下の情報を定義します。

項目名	内容	値	個数
linkplus	係り先の句 ID と、現在の句の ID の差分	数値	1
mci	次句の先頭から、係り先の句の先頭までのモーラ数	数値	1
mcle	次句の先頭から、係り先の句の末尾までのモーラ数	数値	1
lpos	現在の句の最終形態素の品詞名	品詞	２１

数値の範囲を指定したい場合には、".."を使用します。
例えば、"2..4"は「２以上４以下」を示し、"3..*"は「３以上」を示します。
lpos が複数定義された場合 or 条件で判定され、その他は and 条件で判定されます。
論理式で表すと次のようになります。
(linkplus の条件) and (mci の条件) and (mcle の条件) and ((lpos の条件１)
or (lpos の条件 2) ... or (lpos の条件 N))

6．2．4 ipadic.bou

用途：境界情報、および、境界に挿入するポーズの有無や長さについて指定します。また、境界情報と該当品詞の組み合わせを定義します。

ヘッダー部分（BoundarySet/Head）

```
<?xml version="1.0" encoding="x-euc-jp"?>
<BoundarySet>
<Head>
<Name> ipadic </Name>
<Version> 1.00 </Version>
</Head>
 (中略)
 ・
 ・
</BoundarySet>
```

ポーズの挿入有無や長さの指定（BoundarySet/Body/PauseSet）

＜ BoundarySet/Body/PauseSet>に以下のように指定。

```
<Body>
<PauseSet>
   <Pattern name="default">
        <Boundary name="文境界"       always="true"> 1.0  </Boundary>
        <Boundary name="節境界"       always="true"> 0.50 </Boundary>
        <Boundary name="非修飾境界３" always="true"> 0.35 </Boundary>
        <Boundary name="非修飾境界２" always="interval"> 0.21 </Boundary>
        <ToutenPause always="interval"> 0.30 </ToutenPause>
        <IntervalMoras> 15 </IntervalMoras>
        <IntervalLastMoras> 5 </IntervalLastMoras>
   </Pattern>
   <Pattern name="often">
        <Boundary name="文境界"       always="true"> 1.0  </Boundary>
        <Boundary name="節境界"       always="true"> 0.50 </Boundary>
        <Boundary name="非修飾境界３" always="true"> 0.35 </Boundary>
        <Boundary name="非修飾境界２" always="interval"> 0.21 </Boundary>
        <ToutenPause always="true"> 0.30 </ToutenPause>
        <IntervalMoras> 10 </IntervalMoras>
   </Pattern>
</PauseSet>
```

BoundarySet/Body/PauseSet/Pattern

ポーズ挿入のパターンを指定。

name 属性にはパターン名を指定。"default"が、mor2pli でのポーズ挿入のデフォルトパターンとなります。

```
<Pattern   name="default">
```

BoundarySet/Body/PauseSet/Pattern/{Boundary, ToutenPause}

境界情報について、ポーズ挿入規則とポーズ長を指定します。

```
<Boundary name="文境界"        always="true"> 1.0  </Boundary>
<Boundary name="節境界"        always="true"> 0.50 </Boundary>
<Boundary name="非修飾境界３" always="true"> 0.35  </Boundary>
<Boundary name="非修飾境界２" always="interval"> 0.21  </Boundary>
<ToutenPause always="interval"> 0.30 </ToutenPause>
```

Boundary 項目には、name 属性に BoundarySet/List/Boundary で定義した境界情報のうち、ポーズ挿入したいものについて、ポーズ長（単位 秒）を指定します。

name 属性には、境界情報名を指定。

always 属性には、ポーズ挿入規則を指定。

always = { true | interval | false }

true	ポーズを挿入する
interval	ポーズ間のモーラ数に応じて、挿入するかどうかを決定する。
false	ポーズ挿入しない

ToutenPause 項目には、品詞が「記号－読点」の形態素について、ポーズ挿入するかどうかを指定します。読点を無視して境界情報のみでポーズ挿入する場合には、当項目は指定しません。

BoundarySet/Body/PauseSet/Pattern/{IntervalMoras, ToutenPause}

Boundary や ToutenPause の always 属性値を"interval"にした場合に、指定します。

```
<IntervalMoras> 15 </IntervalMoras>
<IntervalLastMoras> 5 </IntervalLastMoras>
```

IntervalMoras には、ポーズ自動挿入を行うポーズ間の最低モーラ数を指定します。
IntervalLastMoras には、ポーズ自動挿入しない文末からのモーラ数を指定します。

Pos 項目に指定できる属性について

Pos 項目は、境界情報パターンの指定に使用する基本となる項目で、品詞名を指定します。

Pos 項目	品詞名
	品詞の alias 名（ただし、Alias 定義では使用不可）

以下の属性も指定して、さらに対象を限定することができます。

base	見出し語
read	ヨミ
type	活用型
form	活用形
repeat	繰り返し（*：０回以上の繰り返し、+：１回以上の繰り返し） ただし、Alias 定義中の Pos 指定には使用不可。

以下の指定文字列では、末尾に"*"を指定して文字列記述を省略できます。

- Pos 項目値
- type 属性値
- form 属性値

品詞の Alias 定義 (BoundarySet/Body/Alias)

<BoundarySet/Body/List>での境界情報パターンを書きやすくするため、品詞の別名を定義します。

```
<Alias>
<PosAlias name="句はじまり (動詞) ">
  <Pos> 動詞-自立                </Pos>
  <Pos> 動詞-非自立              </Pos>
  <PosList>
    <Pos> 名詞-サ変接続          </Pos>
    <Pos base="する"> 動詞-自立 </Pos>
  </PosList>
</PosAlias>

<PosAlias name="句はじまり (形容詞) ">
  <Pos> 形容詞-自立              </Pos>
  <PosList>
    <Pos> 名詞-ナイ形容詞語幹    </Pos>
    <Pos base="ない"> 助動詞     </Pos>
  </PosList>
</PosAlias>
</Alias>
```

PosAlias

品詞の別名を指定します。

Pos 項目には、品詞名を指定。PosList 項目には、Pos のリストを指定。

Pos リストを指定した場合には、複数の品詞の並びを条件とし別名で使用されます。

上記例では、形容詞-自立または、名詞-ナイ形容詞語幹，助動詞"ない"の並びに対して、"句はじまり (形容詞) "という別名が定義されます。

境界情報パターンの指定（BoundarySet/Body/List）

境界情報パターンを指定します。（cabochaset ファイルの[UseCabocha] false の時有効）

```
<Boundary  name="文境界">
<Pattern>
     <Bound> <Pos> 記号-句点     </Pos> </Bound>
  <NextList> <Pos> 自立語        </Pos> </NextList>
</Pattern>
<Pattern>
     <Bound> <Pos base="？"> 記号--一般  </Pos> </Bound>
  <NextList> <Pos> 自立語        </Pos> </NextList>
</Pattern>
<Pattern>
     <Bound> <Pos base="！"> 記号--一般  </Pos> </Bound>
  <NextList> <Pos> 自立語        </Pos> </NextList>
</Pattern>
</Boundary>

<Boundary  name="節境界">
<Pattern>
  <PrevList> <Pos> 感動詞       </Pos> </PrevList>
     <Bound> <Pos> 記号-読点    </Pos>   </Bound>
</Pattern>
<Pattern>
  <PrevList> <Pos> 助詞-接続助詞         </Pos></PrevList>
  <PrevList> <Pos read="ナラ" > 助動詞  </Pos></PrevList>
  <PrevList> <Pos read="タラ" > 助動詞  </Pos></PrevList>
     <Bound> <Pos> 記号-読点             </Pos>   </Bound>
  <NextList> <Pos> 自立語（！動詞・形容詞・形容動詞）    </Pos> </NextList>
</Pattern>
```

Boundary

境界情報のパターンを指定します。

name 属性には、境界情報名を指定します。

BoundarySet/Body/List/Boundary/Pattern

境界情報種別にあたる品詞パターンを指定します。

```
<Pattern>
  <PrevList> <Pos>感動詞</Pos> </PrevList>
     <Bound> <Pos> 記号-読点              </Pos>   </Bound>
  <NextList> <Pos> 名詞--一般 </Pos> </NextList>
</Pattern>
```

PrevList	境界の直前の品詞リストを指定する
Bound	境界となる品詞の情報を指定する
NextList	境界の直後の品詞リストを指定する

6．2．5 ipadic.pos

用途：辞書（ipadic）の品詞・活用型・活用形、XIMERA で使用する統語境界について、名前とコードを定義します。

ヘッダー部分（MorphSet/Head）

```
<?xml version="1.0" encoding="x-euc-jp"?>
<MorphSet>

<Head>
<Name> ipadic </Name>
<Version> 1.00 </Version>
</Head>
```

品詞（MorphSet/Body/PosList）

```
<Body>

<PosList>
<Pos Code="0x01000000"> 名詞-一般 </Pos>
<Pos Code="0x01010000"> 名詞-固有名詞-一般 </Pos>
<Pos Code="0x01010100"> 名詞-固有名詞-人名-一般 </Pos>
<Pos Code="0x01010101"> 名詞-固有名詞-人名-姓 </Pos>
<Pos Code="0x01010102"> 名詞-固有名詞-人名-名 </Pos>
<Pos Code="0x01010200"> 名詞-固有名詞-組織 </Pos>
<Pos Code="0x01010300"> 名詞-固有名詞-地域-一般 </Pos>
<Pos Code="0x01010301"> 名詞-固有名詞-地域-国 </Pos>
<Pos Code="0x01020000"> 名詞-代名詞-一般 </Pos>
<Pos Code="0x01020100"> 名詞-代名詞-縮約 </Pos>
<Pos Code="0x01030000"> 名詞-副詞可能 </Pos>
<Pos Code="0x01040000"> 名詞-サ変接続 </Pos>
    （ 中略 ）
</PosList>
```

活用型 (MorphSet/Body/ ConjugateFormType)

```
<ConjugateTypeList>

<ConjugateType Code="0x0100"> カ変・クル </ConjugateType>
<ConjugateType Code="0x0101"> カ変・来ル </ConjugateType>

<ConjugateType Code="0x0200"> サ変・スル </ConjugateType>
<ConjugateType Code="0x0201"> サ変・ースル </ConjugateType>
<ConjugateType Code="0x0202"> サ変・ーズル </ConjugateType>
    ( 中略 )
</ ConjugateTypeList>
```

活用形 (MorphSet/Body/ ConjugateFormList)

```
<ConjugateFormList>

<ConjugateForm Code="0x0100"> 基本形 </ConjugateForm>
<ConjugateForm Code="0x0101"> 音便基本形 </ConjugateForm>
<ConjugateForm Code="0x0102"> 現代基本形 </ConjugateForm>
<ConjugateForm Code="0x0103"> 文語基本形 </ConjugateForm>
<ConjugateForm Code="0x0104"> 語幹 </ConjugateForm>
    ( 中略 )
</ ConjugateFormList>
```

境界情報 (MorphSet/Body/ ModificationTypeList)

```
< ModificationTypeList>

<ModificationType Code="0x0"> 引用境界   </ModificationType>
<ModificationType Code="0x1"> 係り受け   </ModificationType>
<ModificationType Code="0x2"> 節境界     </ModificationType>
<ModificationType Code="0x3"> 複合語内   </ModificationType>
    ( 中略 )
</ ModificationTypeList>
```

6.2.6 リファレンス

include

```
#include "setbound.h"
```

機能の呼出し

CSBound クラスを使用。

(1) コンストラクタ

CSBound:: CSBound (tracelevel, morphset_type, morphset_dir, boundset_dir);		
機能	初期化と、関連ファイルの読込み	
引数	int tracelevel	デバッグモード（通常は 0）
	const char* morphset_type	辞書の種別（"ipadic"）
	const char* morphset_dir	ipadic.pos の格納ディレクトリパス
	const char* boundset_dir	ipadic.bou の格納ディレクトリパス

(2) コンストラクタの終了確認

bool CSBound:: IsValid()		
機能	オブジェクトが動作可能か確認	
引数	なし	
戻り値	true	正常
	false	エラー

(3) 変換

bool CSBound:: Convert(const string in_str, const string out_str)		
機能	解析	
引数	in_str	番茶出力データ（テキスト）
	out_str	境界情報解析データ（テキスト）
戻り値	true	正常
	false	エラー

６．３ 言語韻律情報作成

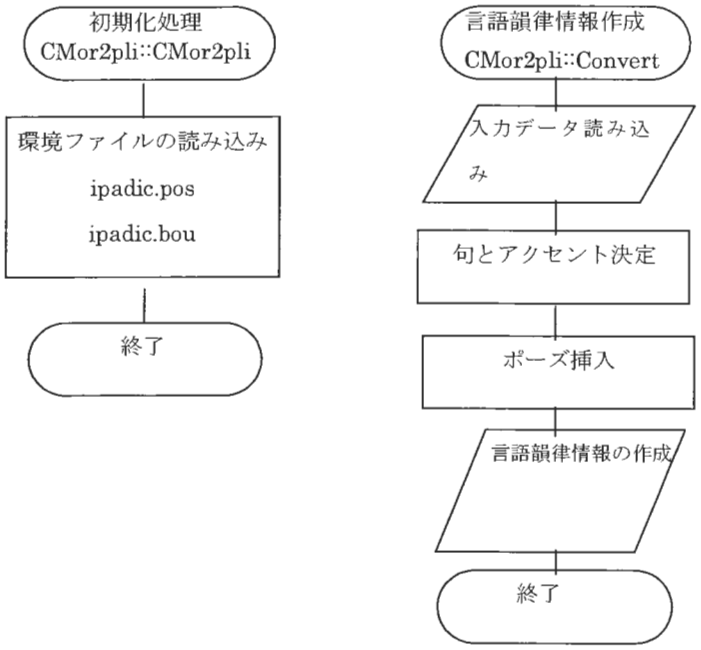
機能

本機能（CMor2Pli クラス）は、形態素解析結果を入力として以下の処理を行い、言語韻律情報を生成します。

- アクセント句の決定
- アクセント句のアクセント核の決定
- 境界情報を元にしたアクセント句間へのポーズ挿入

前提辞書は、ipadic です。

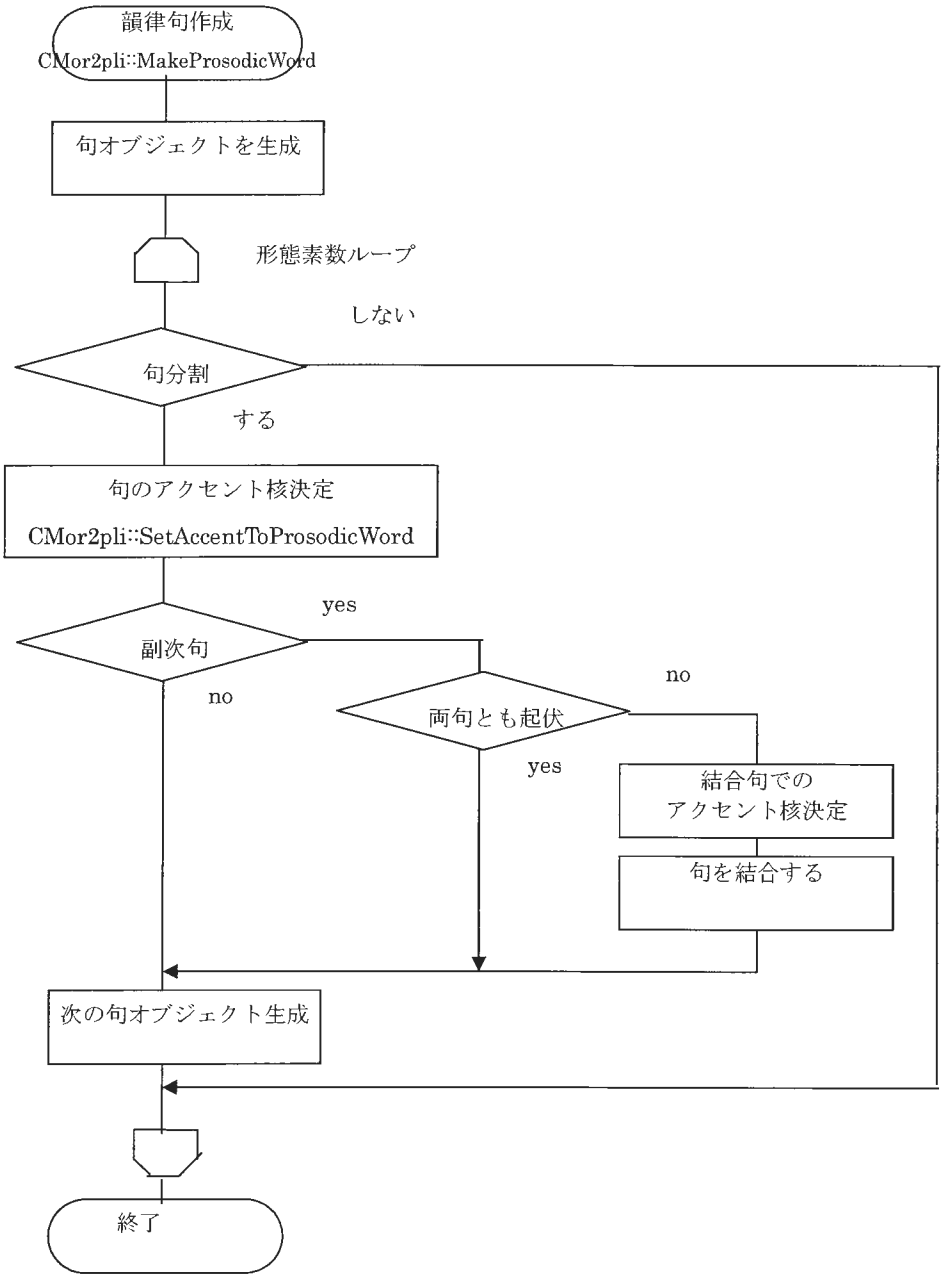
全体の処理の流れ



6．3．1 アクセント句境界と句のアクセント核の位置

文を構成する形態素をチェックして、アクセント句境界と句のアクセント核の位置を決定します。

句とアクセント決定の処理フロー



6．3．1．1 アクセント句境界の決定

形態素の品詞などの情報からアクセント句境界を決定します。

	隣接する形態素		形態素間での句分割		
			する		しない
			句	副次句	
品詞以外の情報で決定	ヨミがない	＊	○		
	境界情報がポーズ必須	＊	○		
	アクセント結合情報がP3（分離型）	＊	○		
	＊	アクセント結合情報がS1（分離型）	○		
	＊	アクセント結合情報がS2（副次アクセント型）		○	
品詞で決定	接頭辞	＊			○
	動詞-連用形	動詞-非自立 形容詞-非自立			○
	名詞-＊ (*1)	名詞-＊ 名詞-接尾辞-＊			○
	名詞-サ変	動詞（する）		○	
	＊	自立語な品詞	○		
	＊	接頭辞	○		
	助詞	！助詞	○		
	動詞 or 助動詞	！付属語な品詞	○		
	動詞 or 形容詞	＊-非自立-＊		○	
	連体詞	＊-非自立-＊		○	
	＊	名詞-非自立-副詞可能		○	
その他					○

(＊1)名詞の連続の場合、固有名詞、数詞など一部の組み合わせに例外があります。

6．3．1．2 句のアクセント核の決定

アクセント句の先頭の形態素から順番にアクセント核の位置を計算し、句のアクセント核を決定します。アクセント核の位置の決定は基本的に形態素のアクセント結合規則に従います。ただし、形態素の組み合わせによっては例外があります。

なお、個々のアクセント結合規則の種類とその規則性は次節「辞書の付加情報」で説明していますので、そちらを参照してください。

(1)句の先頭形態素のアクセント決定

先頭形態素のアクセントを取得する。

先頭形態素に節アクセント結合規則(A1)が定義されており、かつ、前の句がその結合条件を満たす場合、節アクセント結合規則を適用してアクセントを決定する。

(2)アクセント結合

句が複数の形態素から構成される場合、句のアクセント位置を計算する。

・ アクセント結合規則の例外規則

以下の組み合わせにはアクセント結合規則を適用せず例外規則でアクセント核を決める。

形態素の組み合わせ		アクセント核位置	例
助詞	助詞 助動詞	現在のアクセント核が0） 前形態素の最終モーラ	（水に+は） ミズニ+ハ = ミズニ'ハ
		0でない） そのまま	（山に+は） ヤマ'ニ+ハ = ヤマ'ニハ
副詞（平板）	助動詞	前形態素の最終モーラ	（全く+だ） マッタク+ダ = マッタク'ダ
副詞 （繰り返し語）	だ な に	0（平板）にする	（つるつる+に） ツ'ルツル+ニ = ツルツルニ

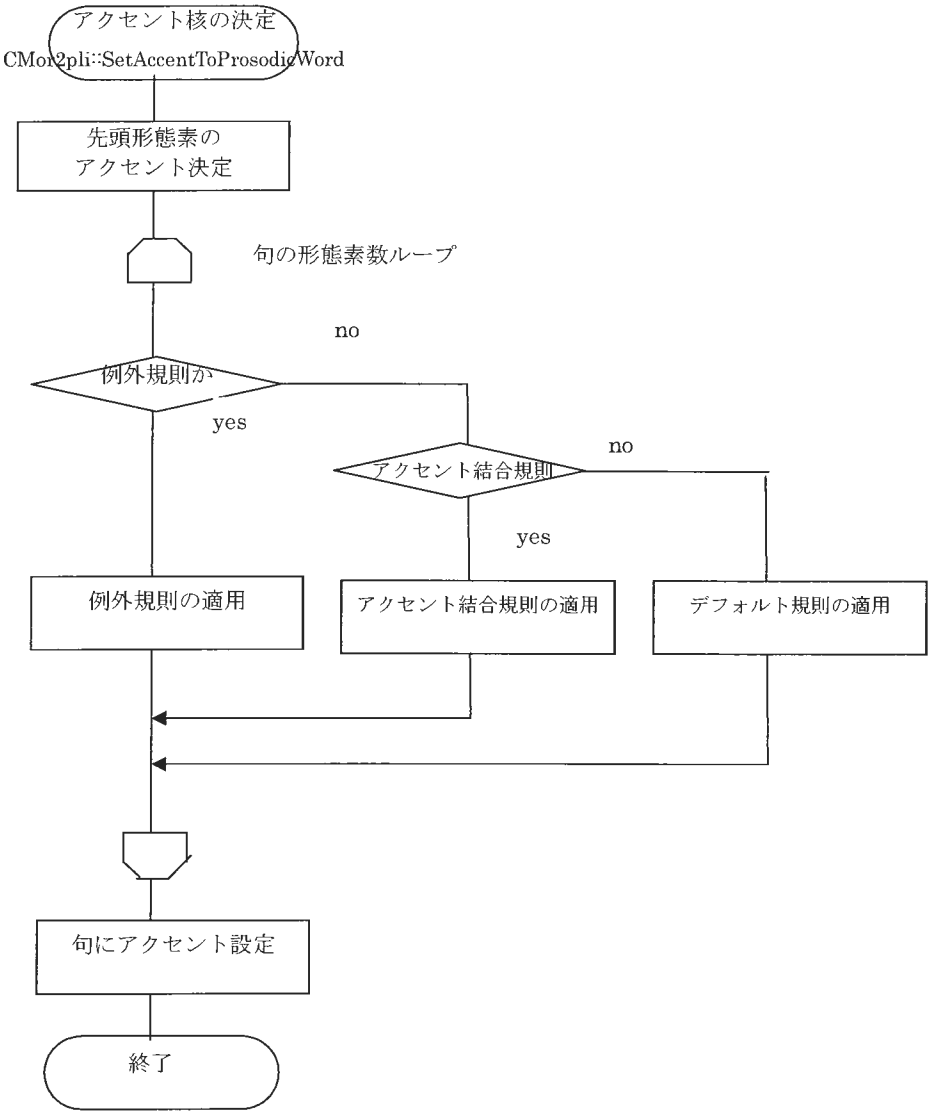
・ アクセント結合規則

形態素の付加情報にアクセント結合規則があり、前形態素とのアクセント結合に適用できる場合、アクセント結合規則を適用してアクセント核を決定する。

・ アクセント結合規則がない

アクセント結合規則がない場合、アクセント結合規則のデフォルト規則でアクセントを決定する。

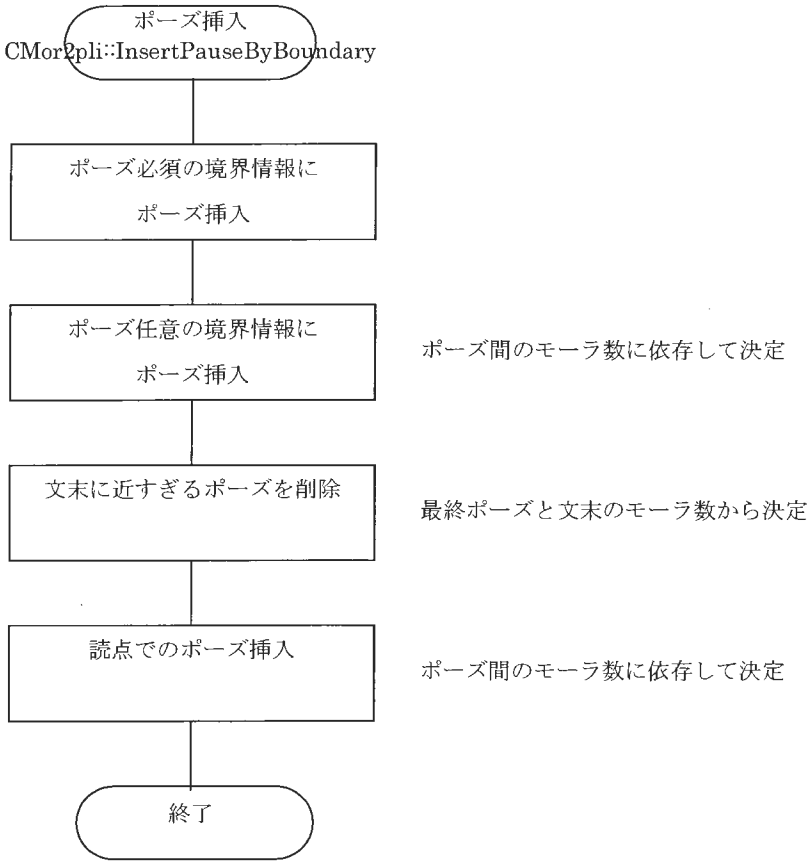
アクセント核の決定



6．3．2 ポーズ挿入

文を構成する各句末にポーズ挿入するかどうかを決定します。判定には、各句末形態素の境界情報のポーズ挿入パターン（ipadic.bou で定義）を使用します。

ポーズ挿入



６．３．３ 辞書の付加情報

mor2pli は辞書の付加情報を元にアクセントを決定します。ここでは、辞書の付加情報について説明します。辞書のその他の項目は ipadic のフォーマットと同様です。

６．３．３．１ 付加情報のフォーマット

付加情報のフォーマットは以下のとおりです。各情報は":"で区切ります。

accent=アクセント情報:accent_con=アクセント結合情報

アクセント情報（accent）

単語のアクセント型を指定します。アクセント核となるモーラの位置（n）と、単語のモーラ数（m）を数字で指定します。区切り文字は"-"です。

アクセント核のモーラ番号－単語の全モーラ数
書式１　accent=n-m
書式２　accent=n1-m1,n2-m2,・・・・

書式１は、１アクセントの単語で使用します。辞書の大部分の単語は書式１です。

書式２は、複数アクセントの単語で使用します。

「ありとあらゆる」のように長い単語の場合、「ア'リト　アラユ'ル」と２アクセントで読んだり、「ア'リトアラユル」と１アクセントで読んだりします。前者は複数アクセントなので書式２"1-3,3-4"と記述します。後者は１アクセントなので書式１"1-7"と記述します。

アクセント結合情報（accent_con）
単語のアクセント結合情報を指定します。

アクセント句が複数の単語で構成される場合、単語のアクセント結合規則に従って、アクセント核の位置を移動させます。アクセント結合規則には、Pn（接頭辞アクセント結合）、Cn（複合名詞アクセント結合）、Fn（付属語アクセント結合）、Sn（句分割アクセント結合）、An（節アクセント結合）があります。それぞれのフォーマットは以下のとおりです。

アクセント結合規則	フォーマット
P1-4（接頭辞アクセント結合）	アクセント結合規則[限定条件]
C1-4（複合名詞アクセント結合）	アクセント結合規則[@アクセント結合価]［限定条件]
F1-6（付属語アクセント結合）	先行品詞%アクセント結合規則［@アクセント結合価］［限定条件]
S1-2（句分割アクセント結合）	アクセント結合規則
A1（節アクセント結合）	先行節%アクセント結合規則@アクセント結合価

[アクセント結合規則]
P1-P4、C1-C4 等があります。それぞれの規則の詳細は次節を参照してください。

[先行品詞]

アクセント結合規則が付属語（FnまたはCFn）の場合、かならず指定します。助詞、助動詞といった付属語は、先行品詞（名詞、動詞、形容詞、特殊助動詞）によって結合規則が異なるためです。

先行品詞 = { 名詞 | 動詞 | 形容詞 | 特殊助動詞 }

「特殊助動詞」とは助動詞の「た」「だ」「です」です。一般に助動詞には「動詞」のアクセント結合規則を使用します。しかし、これらの特殊助動詞に助動詞「う」「た」が結合した場合（「だろう」「だった」等）のアクセント結合規則が「動詞」と異なるため、「特殊助動詞」で別に指定します。

アクセント結合情報（accent_con）

単語のアクセント結合情報を指定します。

アクセント句が複数の単語で構成される場合、単語のアクセント結合規則に従って、アクセント核の位置を移動させます。アクセント結合規則には、Pn（接頭辞アクセント結合）、Cn（複合名詞アクセント結合）、Fn（付属語アクセント結合）、Sn（句分割アクセント結合）、An（節アクセント結合）があります。それぞれのフォーマットは以下のとおりです。

アクセント結合規則	フォーマット
P1-4（接頭辞アクセント結合）	アクセント結合規則[限定条件]
C1-4（複合名詞アクセント結合）	アクセント結合規則[@アクセント結合価]［限定条件]
F1-6（付属語アクセント結合）	先行品詞%アクセント結合規則［@アクセント結合価］［限定条件]
S1-2（句分割アクセント結合）	アクセント結合規則
A1（節アクセント結合）	先行節%アクセント結合規則@アクセント結合価

[アクセント結合規則]

P1-P4、C1-C4 等があります。それぞれの規則の詳細は次節を参照してください。

[先行品詞]

アクセント結合規則が付属語（FnまたはCFn）の場合、かならず指定します。助詞、助動詞といった付属語は、先行品詞（名詞、動詞、形容詞、特殊助動詞）によって結合規則が異なるためです。

先行品詞 = { 名詞 | 動詞 | 形容詞 | 特殊助動詞 }

「特殊助動詞」とは助動詞の「た」「だ」「です」です。一般に助動詞には「動詞」のアクセント結合規則を使用します。しかし、これらの特殊助動詞に助動詞「う」「た」が結合した場合（「だろう」「だった」等）のアクセント結合規則が「動詞」と異なるため、「特殊助動詞」で別に指定します。

［先行節］

「人」「日」「上」のように、単独で読む場合のアクセントと、連体修飾された場合のアクセントが異なる単語で使用します。

先行節　＝　｛ 連体修飾節 ｝

［結合アクセント価］

結合アクセント価は、アクセント結合規則を適用した場合のアクセント位置を示します。数字で指定します。

通常は正の値ですが、形容詞のときなどに負の値になることがあります。これは形態素の結合位置を起点（０）として、アクセントが前（負）後（正）のいずれにずれるかを指定するものです。

例えば、接続助詞「ば」の結合アクセント価は形容詞%F2@-2 となっており、これが平板型の形容詞に下接する場合は、「アツケレ」＋「バ」で「バ」の前を起点（０）として、-2 ですので２モーラ前にずれ、「ケレ」の前のモーラにアクセント核がつくことになります。

結合アクセント価が省略された場合は、"accent="に指定されたアクセント核の値を使用します。

［限定条件］

接尾辞などで、先行形態素のモーラ数によってアクセント結合規則を切り替えたい場合に使用します。

限定条件　＝　｛　le　｜　eq　｜　ge　｝前形態素のモーラ数

le は、"less [than] or equal [to]"の省略で、「以下」にあたります。

eq は、"equal"の省略で、「等しい」にあたります。

ge は、"greater [than] or equal [to]"の省略で、「以上」にあたります。

なお、この限定条件を付加した場合、この条件にあてはまらないときの結合規則も必要となります。以下のように、"else"でつなげて、記述します。

（例）　accent_con=C4eq2*else*C3

6．3．3．2 アクセント結合規則

(1) 1つの句で完結するもの

Pn（接頭辞アクセント結合）、Cn（複合名詞アクセント結合）、Fn（付属語アクセント結合）は、1つの句の中で隣接する単語について適用するアクセント結合規則です。
ここでは、結合すべき単語を以下のように定義して、結合規則を説明します。

N_1モーラM_1型 ＋ N_2モーラM_2型$\tilde{M}_2$価 ＝ N_CモーラM_C型

Nは形態素のモーラ数です。
Mは"accent="に指定されたアクセント型です。
$\tilde{M}$は"accent_con=F2@1"のように@の後に指定された結合アクセント価です。
結合アクセント価がない場合（@以降がない）は、M を $\tilde{M}$ として使用します。

[接頭辞アクセント結合]

結合様式	文節のアクセント型（M_C）	
	M_2 = 0 or N_2*（平板 or 尾高）	M_2 ≠ 0 or N_2*
P1 （一体化型）	0	N_1 ＋ M_2
P2 （自立語結合型）	N_1 ＋ 1	N_1 ＋ M_2
P3 （分離型）	M_1	
P4 （混合型）	N_1	M_2

(*)後続名詞の最終音節が2モーラからなる（最終モーラが特殊拍）場合、N_2をN_2－1にします。

[複合名詞アクセント結合]

結合様式	文節のアクセント型（M_C）
C1 （保存型）	N_1 ＋ $\tilde{M}_2$
C2 （生起型）	N_1 ＋ 1
C3 （標準型）	N_1 （N_1最終モーラが特殊拍または重母音なら、N_1 － 1）
C4 （平板化型）	0

[付属語アクセント結合]

結合様式	文節のアクセント型（M_C）	
	$M_1 = 0$	$M_1 \neq 0$
F1 （従属型）	M_1	
F2 （不完全支配型）	$N_1 + \tilde{M}_2$	M_1
F3 （融合型）	M_1	$N_1 + \tilde{M}_2$
F4 （支配型）	$N_1 + \tilde{M}_2$	
F5 （平板化型）	0	
F6 （「の」特殊型）	M_1 → ただしM_1が尾高の場合は、平板にする	

(*)F5 に該当する単語は辞書にありません。該当する可能性のある単語は助詞の「だけ」だけですが、これは F1 でもよいため辞書では F1 としています。

[デフォルト規則]

適用すべきアクセント結合規則がない場合、デフォルト規則として以下を適用します。

結合様式	文節のアクセント型（M_C）	
	$M_1 = 0$	$M_1 \neq 0$
アクセント結合規則がない	M_2	M_1

(*)形態素解析処理で未知語とされた文字列には、付加情報が存在しないため、上記規則を適用します。

(2)前の句との関係があるもの

Sn（句分割アクセント結合）は、指定した単語を句の先頭として扱い、前の句とどのような結合を行うかを指定する結合規則です。

[句分割アクセント結合]

<table>
<tr><td colspan="3" rowspan="2">結合様式</td><td colspan="2">自句</td></tr>
<tr><td>平板</td><td>起伏</td></tr>
<tr><td colspan="3">S1　（独立アクセント句）</td><td colspan="2">別の句</td></tr>
<tr><td rowspan="2">S2</td><td rowspan="2">（副次アクセント句）</td><td rowspan="2">前の句</td><td>平板</td><td rowspan="2">結合。
アクセントは前の句</td><td>結合。
アクセントは自句</td></tr>
<tr><td>起伏</td><td>別の句</td></tr>
</table>

(*)S1 に該当する単語は辞書にありません。

(*)S2 は助動詞の「ござる」で使用しています。結合例は以下のとおり。

前の句が平板なら句を結合します（例：「申し訳ございません」）。

モウシワケ ｜ ゴザイマセ'ン　＝　モウシワケゴザイマセ'ン

前の句が起伏なら句はそのままです（例：「ありがとうございます」）。

アリ'ガトウ ｜ ゴザイマ'ス

An（節アクセント結合）は、前の句の種別やアクセント型で指定した単語のアクセントを変更する結合規則です。

[節アクセント結合]

結合様式	文節のアクセント型（M_C）	
	$M_1 = 0$	$M_1 \neq 0$
A1	$\tilde{M}_2$	M_2

(*)辞書で A1 を指定した単語は、「人」「日」「上」「下」「所」です。

６．３．４ 入力フォーマット

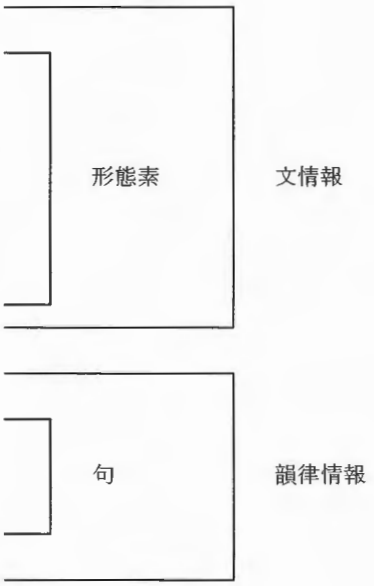
境界情報解析の出力フォーマットです。境界情報解析の出力フォーマットを参照してください。

6．3．5 出力フォーマット

言語韻律情報を XML 形式で出力します。

```
<?xml version="1.0" encoding="Shift_JIS"?>
<PLInfo>
<Head>
<Author Date="2004/3/11"> mor2pli</Author>
</Head>

<Body>
    <sentence  MorphSet=" ipadic" >
        <Morph   Pos=品詞  Kanji=入力文字
            BoundaryType=境界情報
            ConjugateType=活用型
            ConjugateForm=活用形>
            読み
        </ Morph >
    </sentence>
    <Utterance>
        <ProsodicWord  Accent=アクセント位置>
            <Mora   Kana=読み>発音</Mora >
        </ProsodicWord >
    </Utterance >
</Body>
</PLInfo>
```



言語韻律情報サンプル

```
<?xml version="1.0" encoding="Shift_JIS"?>
<PLInfo>
<Head>
<Author Date="2004/3/18"> mor2pli</Author>
</Head>

<Body>

  <Sentence MorphSet="ipadic">
    <Morph Pos="名詞-一般" Kanji="救急車">キュウキュウシャ</Morph>
    <Morph Pos="助詞-格助詞-一般" Kanji="が">ガ</Morph>
    <Morph Pos="名詞-形容動詞語幹" Kanji="十分">ジュウブン</Morph>
    <Morph Pos="助詞-副詞化" Kanji="に" BoundaryType="係り受け">ニ</Morph>
    <Morph Pos="動詞-自立" Kanji="動け" ConjugateType="一段" ConjugateForm="未然形">ウゴケ</Morph>
    <Morph Pos="助動詞" Kanji="ず" ConjugateType="特殊・ヌ" ConjugateForm="連用ニ接続" BoundaryType="非修
飾境界３">ズ</Morph>
    <Morph Pos="名詞-サ変接続" Kanji="救助" BoundaryType="複合語内">キュウジョ</Morph>
    <Morph Pos="名詞-サ変接続" Kanji="作業">サギョウ</Morph>
    <Morph Pos="助詞-格助詞-一般" Kanji="が" BoundaryType="係り受け">ガ</Morph>
    <Morph Pos="動詞-自立" Kanji="遅れ" ConjugateType="一段" ConjugateForm="連用形">オクレ</Morph>
    <Morph Pos="助詞-接続助詞" Kanji="て">テ</Morph>
    <Morph Pos="動詞-非自立" Kanji="いる" ConjugateType="一段" ConjugateForm="基本形">イル</Morph>
  </Sentence>

  <Utterance>
    <ProsodicWord Accent="3">
      <Mora Kana="キュ">キュ</Mora>
      <Mora Kana="ウ">ウ</Mora>
      <Mora Kana="キュ">キュ</Mora>
      <Mora Kana="ウ">ウ</Mora>
      <Mora Kana="シャ">シャ</Mora>
      <Mora Kana="ガ">ガ</Mora>
    </ProsodicWord>
    <ProsodicWord Accent="3">
      <Mora Kana="ジュ">ジュ</Mora>
      <Mora Kana="ウ">ー</Mora>
      <Mora Kana="ブ">ブ</Mora>
      <Mora Kana="ン">ン</Mora>
      <Mora Kana="ニ">ニ</Mora>
    </ProsodicWord>
    <ProsodicWord Accent="0" Pause="0.35">
      <Mora Kana="ウ">ウ</Mora>
      <Mora Kana="ゴ">ゴ</Mora>
      <Mora Kana="ケ">ケ</Mora>
      <Mora Kana="ズ">ズ</Mora>
    </ProsodicWord>
    <ProsodicWord Accent="4">
      <Mora Kana="キュ">キュ</Mora>
(以下、省略)
```

読みのある形態素のみ出力。
句読点など読みのないものは出力しません

句末にポーズ挿入
単位は標準話速の比率。

6.3.6 リファレンス

include

```
#include "mor2pli.h"
```

機能の呼出し

CMor2Pli クラスを使用。

(1)コンストラクタ

CMor2Pli::CMor2Pli(tracelevel, morphset_type, morphset_dir);		
機能	mor2pli の初期化と、関連ファイルの読込み	
引数	int tracelevel	デバッグモード（通常は 0）
	const char* morphset_type	辞書の種別（"ipadic"）
	const char* morphset_dir	環境ファイル（ipadic.pos, ipadic.bou）の格納ディレクトリパス

(2)コンストラクタの終了状態の確認

bool Cmor2Pli:: IsValid()		
機能	mor2pli が動作可能か確認	
引数	なし	
戻り値	true	正常
	false	エラー

(3)変換

bool CMor2Pli:: Convert(const string instr, const string outstr)		
機能	解析	
引数	instr	境界情報解析データ（テキスト）
	outstr	言語韻律情報（テキスト）
戻り値	true	正常
	false	エラー

６．４ 拡張環境依存ラベル変換

本モジュールは、言語韻律情報を HTS の入力形式に変換します。

このモジュールは、下記技術資料に基づいて実装されています。
全，陸，倪，徳田，河井，“日本語及び中国語音声合成のための韻律生成部の設計”，ATR テクニカルレポート（非公開）TR-SLT-0032，2002 年 12 月

このモジュールは xdata/settings/linux or windows の位置にある、ATR_jp_00.kp、devoice.def ファイルを使用してカナ文字から音素名、無声化ルールを定義しています。

6．4．1 ATR_jp_00.kp

片仮名を音素へ変換するためのルールテーブルです。
ファイルは XML 形式で記述されています。

フォーマット

<PhonemeDef>	音素リストの定義	
</PhonemeDef>	Silence	文頭文末の無音記号
	Pause	文中の無音記号
<Phoneme> </Phoneme>	音素記号	
<Kana2PhonemeTable>	片仮名を音素に変換する表	
</Kana2PhonemeTable>		
<Kana2Phoneme>	対応する音素列	
</Kana2Phoneme>	Kana	片仮名文字

encoding 属性は windows のとき"shift_jis"、Linux のとき"x-euc-jp"と表記しています。

ファイル内容

```
<?xml version="1.0" encoding="shift_jis"?>
<KanaAndPhoneme>

<Head>
<Name> ATR_jp_00 </Name>
<Version> 1.00 </Version>
</Head>

<Body>
<PhonemeDef Silence="sil" Pause="pau">
<Phoneme> sil </Phoneme>
<Phoneme> Q </Phoneme>
<Phoneme Devoiced="I" Elongated="e"> i </Phoneme>
<Phoneme Devoiced="E"> e </Phoneme>
```

```
<Phoneme Devoiced="A"> a </Phoneme>
<Phoneme Devoiced="O"> o </Phoneme>
<Phoneme Devoiced="U" Elongated="o"> u </Phoneme>
<Phoneme voiced="i"> I </Phoneme>
<Phoneme voiced="e"> E </Phoneme>
<Phoneme voiced="a"> A </Phoneme>
<Phoneme voiced="o"> O </Phoneme>
<Phoneme voiced="u"> U </Phoneme>
<Phoneme> w </Phoneme>
<Phoneme> y </Phoneme>
<Phoneme> m </Phoneme>
<Phoneme> n </Phoneme>
<Phoneme> N </Phoneme>
<Phoneme> my </Phoneme>
<Phoneme> ny </Phoneme>
<Phoneme> r </Phoneme>
<Phoneme> ry </Phoneme>
<Phoneme> p </Phoneme>
<Phoneme> t </Phoneme>
<Phoneme> k </Phoneme>
<Phoneme> py </Phoneme>
<Phoneme> ky </Phoneme>
<Phoneme> b </Phoneme>
<Phoneme> d </Phoneme>
<Phoneme> g </Phoneme>
<Phoneme> by </Phoneme>
<Phoneme> dy </Phoneme>
<Phoneme> gy </Phoneme>
<Phoneme> s </Phoneme>
<Phoneme> sh </Phoneme>
<Phoneme> h </Phoneme>
<Phoneme> f </Phoneme>
<Phoneme> hy </Phoneme>
<Phoneme> z </Phoneme>
<Phoneme> j </Phoneme>
<Phoneme> ch </Phoneme>
```

```
<Phoneme> ts </Phoneme>
<Phoneme Merge="left"> > </Phoneme>
<Phoneme Merge="right"> cl </Phoneme>
<Phoneme > pau </Phoneme>
</PhonemeDef>

<Kana2PhonemeTable>
<Kana2Phoneme Kana="ア"> a </Kana2Phoneme>
<Kana2Phoneme Kana="イ"> i </Kana2Phoneme>
<Kana2Phoneme Kana="ウ"> u </Kana2Phoneme>
<Kana2Phoneme Kana="エ"> e </Kana2Phoneme>
<Kana2Phoneme Kana="オ"> o </Kana2Phoneme>
<Kana2Phoneme Kana="カ"> k a </Kana2Phoneme>
<Kana2Phoneme Kana="キ"> k i </Kana2Phoneme>
<Kana2Phoneme Kana="ク"> k u </Kana2Phoneme>
<Kana2Phoneme Kana="ケ"> k e </Kana2Phoneme>
<Kana2Phoneme Kana="コ"> k o </Kana2Phoneme>
...
 (途中略)
...
</Kana2PhonemeTable>
</Body>
</KanaAndPhoneme>
```

6.4.2 devoice.def

音素の無声化ルールを定義しています。
ファイルはXML形式で記述されています。

フォーマット

<Rule> </Rule>		
<Syl> </Syl>	無声化対象音節の 子音部 同母音部	
<If> </If>	後続音節の子音部 同母音部	後続音節の母音部=*の場合・・・任意の母音
<Then> </Then>	アクション	yes 無声化可能 no 無声化しない always 必ず無声化する yes_exceptfor_accnuc アクセント核がなければ無声化可能 yes_exceptfor_accnuc_wordinit アクセント核がなくかつ語頭でなければ無声化可能 yes_exceptfor_wordinit 語頭でなければ無声化可能 yes_for_accbase アクセント核より後ろにあれば無声化可能

ファイルの内容

```
<?xml version="1.0" encoding="shift_jis"?>
<DevoiceRules>

<Head>
<Name> ATR_jp_00 </Name>
<Version> 1.00 </Version>
</Head>

<Body>

<Rule> <Syl> k i </Syl>
    <If> Q   *  </If> <Then> yes </Then>
    <If> pau *  </If> <Then> yes </Then>
    <If> sil *  </If> <Then> yes </Then>
    <If> k   *  </If> <Then> yes </Then>
    <If> ky  *  </If> <Then> yes </Then>
    <If> s   *  </If> <Then> yes </Then>
    <If> sh  *  </If> <Then> yes </Then>
    <If> t   *  </If> <Then> yes </Then>
    <If> ch  *  </If> <Then> yes </Then>
    <If> ts  *  </If> <Then> yes </Then>
    <If> h   *  </If> <Then> yes_exceptfor_accnuc </Then>
    <If> hy  *  </If> <Then> yes_exceptfor_accnuc </Then>
    <If> f   *  </If> <Then> yes_exceptfor_accnuc </Then>
    <If> p   *  </If> <Then> yes_exceptfor_accnuc </Then>
</Rule>
...
 (途中略)
...

</Body>
</DevoiceRules>
```

encoding 属性は windows のとき"shift_jis"、Linux のとき"x-euc-jp"と表記しています。

6.4.3 リファレンス

include

```
#include "pli2lab.h"
```

機能の呼出し

CPli2Lab クラスを使用。

(1)コンストラクタ

CPli2Lab::CPli2Lab (tracelevel, kana2PhmSetPath, morphset_dir, devoiceRuleFile);		
機能	初期化と、関連ファイルの読込み	
引数	int tracelevel	デバッグモード（通常は 0）
	const char* kana2PhmSetPath	ATR_jp_00.kp ファイル名
	const char* morphset_dir	ipadic.pos の格納ディレクトリパス
	const char* devoiceRuleFile	devoice.def ファイル名

(2)コンストラクタの終了確認

bool CPli2Lab::IsValid()		
機能	オブジェクトが動作可能か確認	
引数	なし	
戻り値	true	正常
	false	エラー

(3)変換

string CPli2Lab::Convert(const string in_str)		
機能	解析	
引数	in_str	言語韻律情報（テキスト）
戻り値	string	拡張環境依存ラベル

６．５ 素片選択・接続処理の概要

６．５．１ 処理の流れ

韻律予測（HTS）から取得した音素長、F0、パワー、メルケプストラムをもとに最適な素片をデータベースより検索し、素片接続を行います。
素片選択する条件は、サブコスト関数より得られたコスト値の重み付け和が最も小さいコストを最適とします。サブコスト関数は以下の特徴量を使用します。

- F0 の違いに関するサブコスト（ SC_{F0} ）
- 音素持続時間の違いに関するサブコスト（ SC_{dur} ）
- 平均スペクトルの違いに関するサブコスト（ SC_{cen}）
- 音素環境代替に関するサブコスト（ SC_{env} ）
- スペクトルの不連続に関するサブコスト（ SC_{spg} ）
- F0 不連続性に関するサブコスト（ SC_{F0c} ）

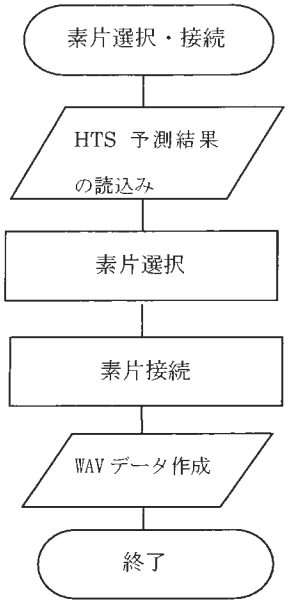
素片のコストは、サブコストをターゲットコスト、接続コストの二つのグループに分け重み付けを行い、素片系列におけるターゲットコストのノルムと接続コストのノルムの重み付け和を統合コストとして求めます。
さらにコスト計算では、音素単位をダイフォン単位に分割し、最適な素片選択を行います。

※参考文献

戸田,河井,津崎,鹿野,“素片接続型日本語テキスト音声合成における音素単位とダイフォン単位に基づく素片選択”，電子情報通信学会論文誌，D-Ⅱ，Vol.J85-D-Ⅱ，No.12，pp.1760-1770，2002 年 12 月

戸田智基，河井 恒，津崎 実，“波形接続型音声合成における知覚的評価に基づく素片選択サブコスト関数の最適化”，電子情報通信学会技術報告，SP2003-81，pp.43-48，2003

素片選択・接続　処理フロー



HTS 予測結果の読み込み

音素長、F0、パワー、メルケプストラムを読み込みます。

素片選択

HTS 予測結果の音素名からデータベースに存在する音素へ変換し、変換された音素をダイフォン単位へ分割します。分割された素片のコスト値から最適な素片系列を検索します。

素片接続

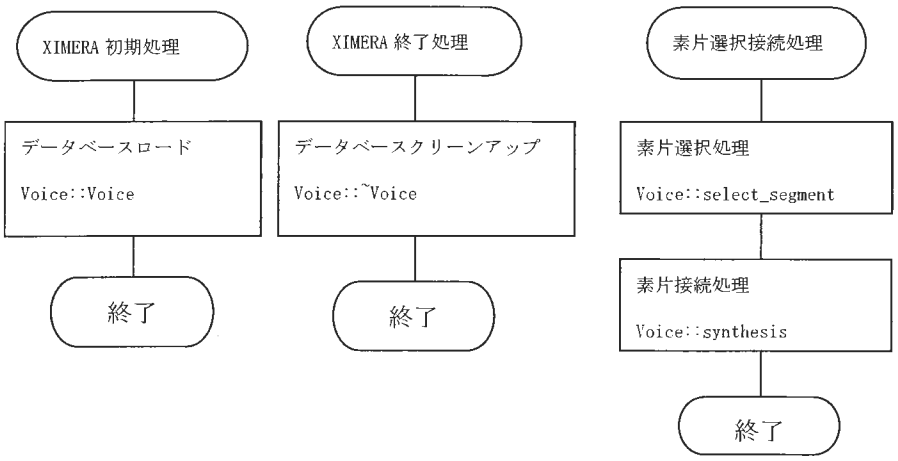
単位選択された素片に該当する音声データを音声ファイルから取得し、接続します。

６．５．２ 素片選択・接続クラス

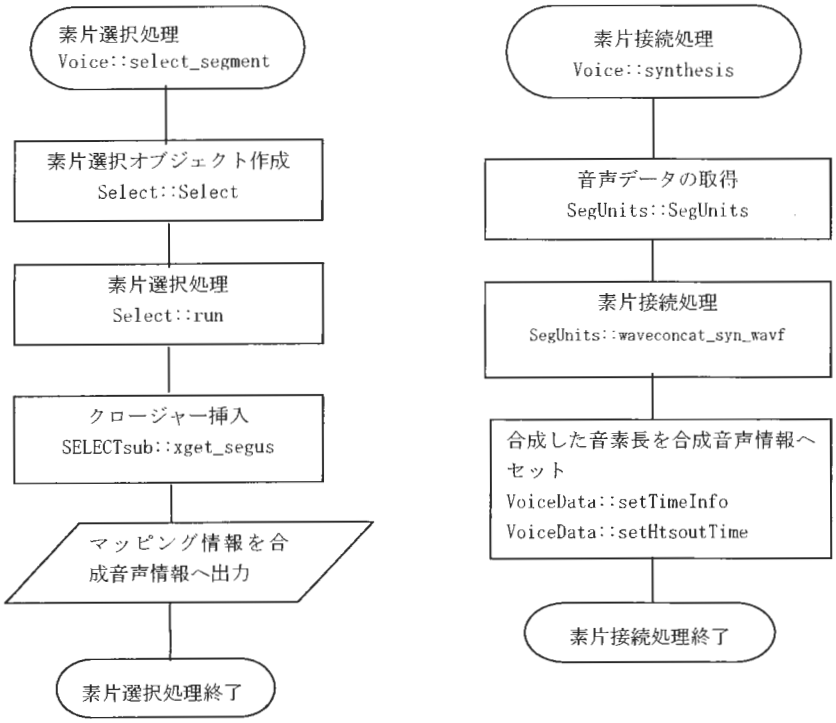
素片選択・接続処理はVoiceクラスで管理されています。Voiceインスタンスを作成することでデータベースがロードされ、Voiceインスタンス削除でデータベースがクリーンアップされます。素片選択・接続におけるXIMERA APIは以下のようにVoiceクラスのメソッドをコールしています。

```
ximera_speaker()        -> voice = new Voice( dbDir )
ximera_cleanup()        -> delete voice
ximera_synth_select()   -> voice->select_segment( htsout, vd )
ximera_synth_connect()  -> voice->synthesis( vd, wavefile )
```

Voiceクラス



素片選択処理、素片接続処理の結果はグローバル変数としてVoiceDataクラスのインスタンスで保持され、合成処理前にクリーンアップされます。



＊合成音声情報は、処理結果を保持する VoiceData クラスです。

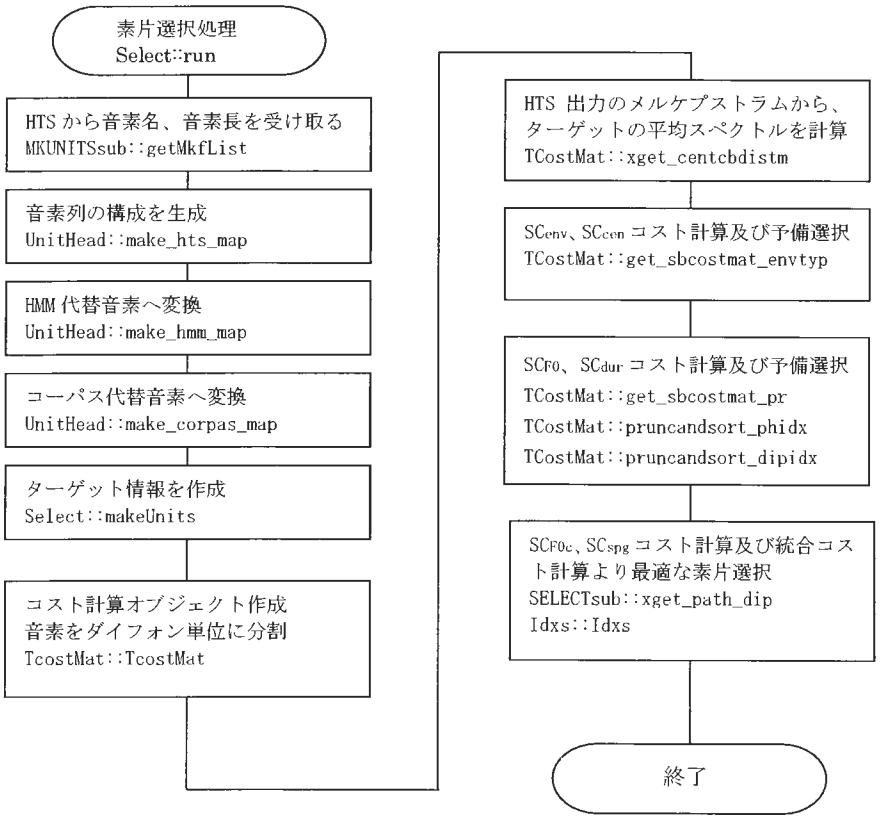
６．６ 素片選択処理

素片選択では、HTS 予測結果を元にデータベースより最適な素片候補を検索します。素片選択処理はまずHMM 代替音素マッピング、コーパス代替マッピングを行い、HTS から出力された音素名をデータベースに存在する音素名へ変換します。その後、変換後の音素名・音素長データと HTS 予測結果の F0、パワー情報を用い、合成するターゲット情報を作成します。そのターゲット情報（音素単位）を、ダイフォン単位へ分割し、コスト計算を行います。

コスト計算処理では、素片候補が大量に存在するため、一部のサブコスト関数を計算した時点で、素片候補を絞り込みます（予備選択)。絞り込み後、サブコスト値が最小となる最適な素片を選択します。最後に、選択された素片へクロージャー挿入を行います。

※クロージャー：破裂（破擦）音に伴う閉鎖区間および促音に伴う休止区間。

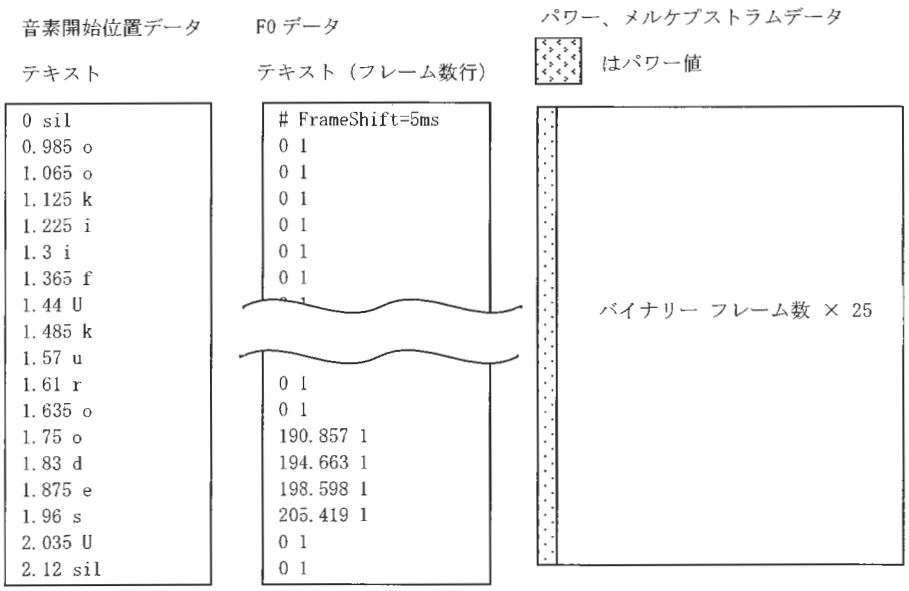
素片選択処理フロー



6.6.1 HTS 予測結果

HTS 予測結果は、音素開始位置、F0、メルケプストラムとパワーの３つのデータから構成されています。音素開始位置は、音素開始時間と対応する音素名のテキストデータ、F0 データは、5ms 単位の logF0 のテキストデータ、パワー、メルケプストラムデータは、5ms 単位のパワー（log10(power)）と 5ms 単位、24 要素から成るメルケプストラムで構成されています。

HTS 予測結果サンプル



音素名は、データベース上に存在する音素に変換され、素片候補の選別に使用されます。音素長は音素持続時間の違いに関するサブコストの計算等に使用されます。

・ F0 は、予測結果と素片間の F0 誤差に関するサブコスト等の計算に使用されます。

パワーは、素片接続時に音声ファイルとターゲット情報の間のパワー差を調整したり、サブコストを計算するために使用されます。

メルケプストラムは、平均スペクトルの誤差に関するサブコストの計算に使用されます。

６．６．２ 代替音素のマッピング

代替音素のマッピングでは、HTS 予測結果の音素名をデータベースに存在する音素へ変換します。

HMM 代替音素の変換テーブルは hmmphmap.txt で定義され、この情報を元に変換を行います。変換テーブルは、変換元 n、変換先１の n 対１のルールで変換します。変換時、音素長も合計します。

コーパス代替音素の変換テーブルは phdefmap.txt で定義され、この情報よりデータベースに存在する音素へ変換します。変換テーブルは、変換元、変換先１対１のルールで変換します。

音素名変換例

上段は音素名、下段は音素長(ms)

HTS 予測結果

sil	o	o	k	i	i	f	U	k	u	r	o	o	d	e	s	U	sil
985	80	60	100	75	65	75	45	85	40	25	115	80	45	85	75	85	955

HMM マッピング

sil	oo	k	ii	fU	k	u	r	oo	d	e	sU	sil
985	140	100	140	120	85	40	25	195	45	85	160	955

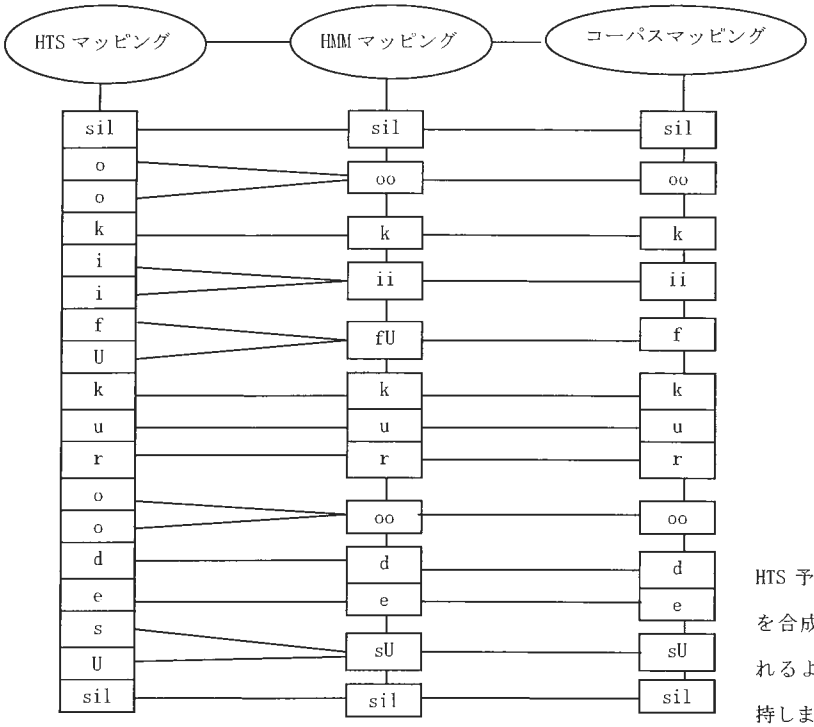
コーパス代替マッピング

sil	oo	k	ii	f	k	u	r	oo	d	e	sU	sil
985	**140**	**100**	**140**	**120**	**85**	**40**	**25**	**195**	**45**	**85**	**160**	**955**

※コーパスで fU の音素が指定数より少ない場合

マッピング情報は、代替音素マッピングの経過を保持するため、VoiceData オブジェクトへ格納されます。
下図は、VoiceData オブジェクトで保持するマッピング構成イメージです。

音素マッピング構成イメージ



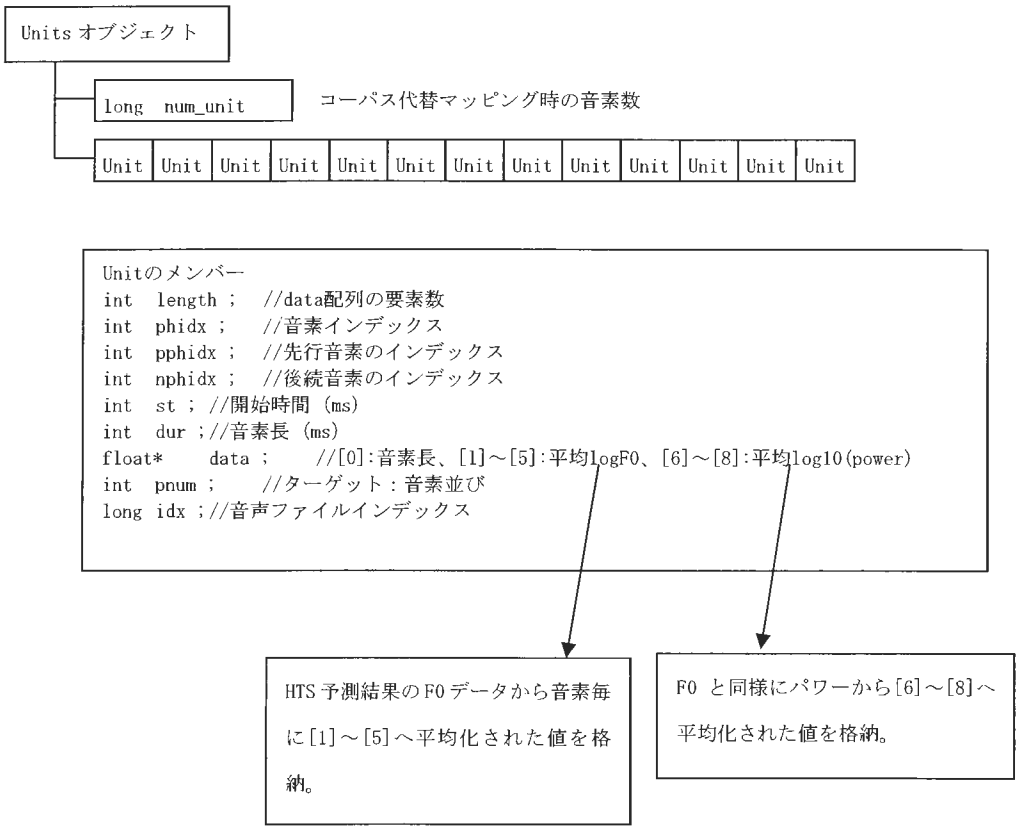
HTS 予測結果の音素単位を合成データからたどれるよう変換経過を保持します。

6.6.3 ターゲット情報

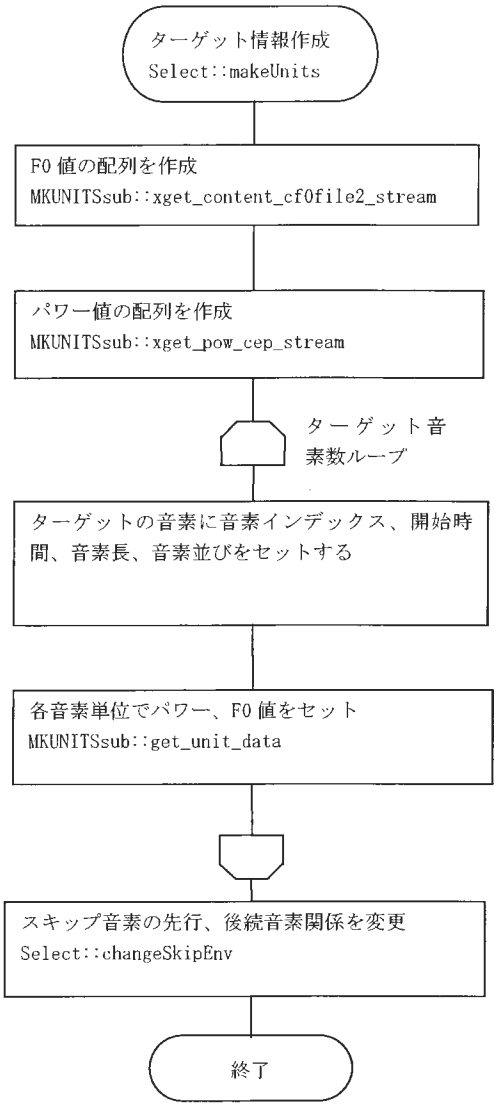
ターゲット情報は Units クラスで管理され、コーパス代替音素へ変換された音素情報を元にオブジェクトが構成されます。

各音素の情報が格納される Unit クラスは、素片候補のデータベースと同じクラスで管理されています。

ターゲット情報構成



ターゲット情報作成フロー



HTS 予測結果のメルケプストラムは、シフト長 5ms の 24 要素数で構成されています。

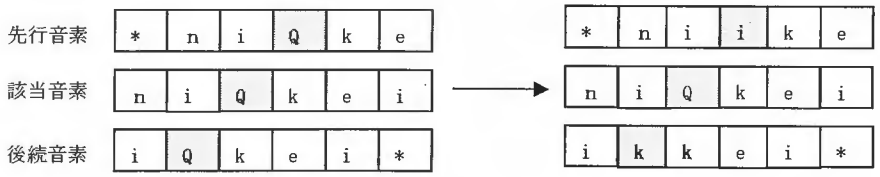
パワーは配列の先頭に格納されています。

6.6.1 HTS 予測結果を参照

ターゲット情報生成後、Units::changeSkipEnv 関数で skipph.txt に定義された音素を検索し、先行、後続音素の指定を変更します。下の例では Q がスキップ音素に指定されている場合です。入力音素列 niQkei において k の先行音素 Q が Q の先行音素 i に i の後続音素 Q が Q の後続音素 k にそれぞれ置き換えられます。

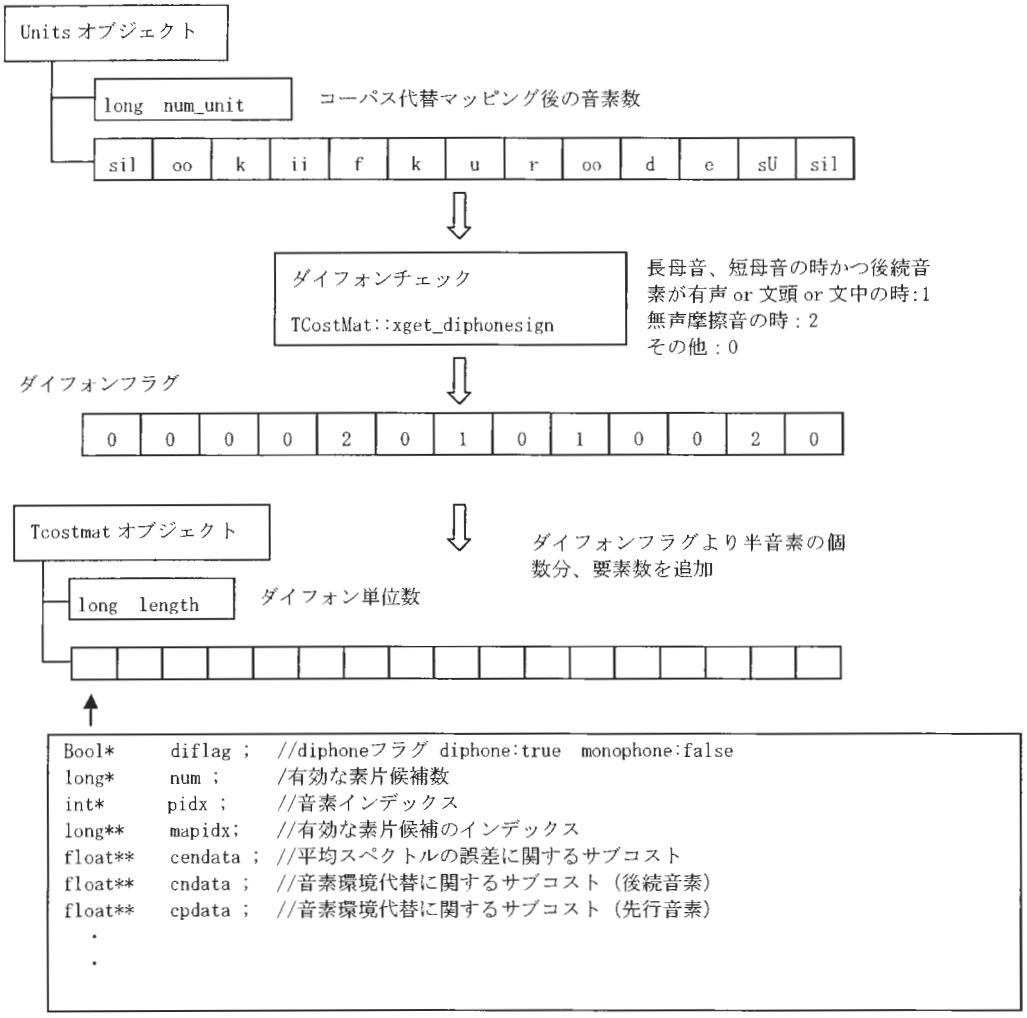
例

Q がスキップ音素

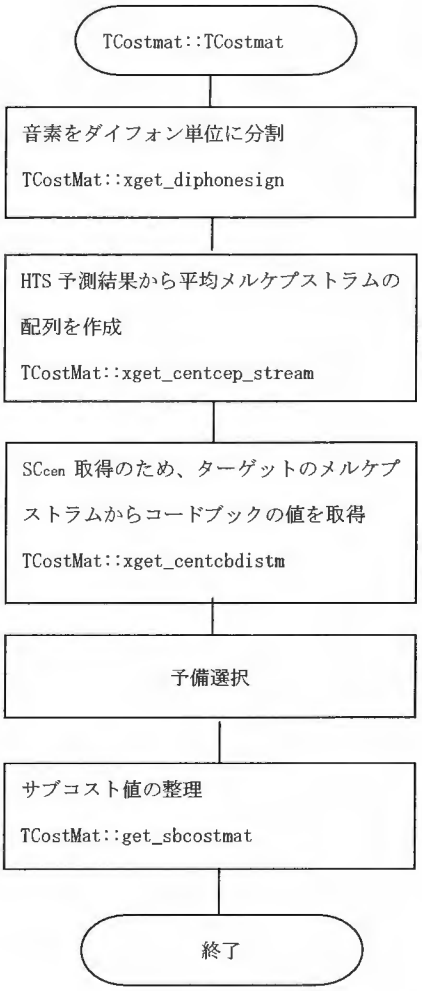


6.6.4 ターゲットコストの計算

TCostMat オブジェクトは、音素単位のターゲット情報をダイフォン単位に分割し、ターゲット音素に対する素片候補およびターゲットコスト値を格納します。各コスト値取得後、素片候補に対して予備選択が行われます。



TcostMat コンストラクタ処理フロー



TCostMat::xget_centcep_stream、
TCostMat:: xget_centcbdistm
は SCcen コストを取得するための準備です。

予備選択については 6.6.5 を参照してください

サブコスト関数の重み付け

サブコスト関数は６つから構成され、ターゲットコスト、接続コストの２つに分け重み付けを行っています。

ターゲットコスト：SC_{cen}、SC_{F0}、SC_{dur}

接続コスト：SC_{env}、SC_{F0c}、SC_{spg}

各素片の統合コストは以下の計算で求めます。

tarp=1.0
conp=1.5
conw=0.7
tarw=1.0-conw

ターゲットコスト
$W_{cen} + W_{dur} + W_{F0} = 1$　　（重み）
$TC = W_{cen} * SC_{cen} + W_{dur} * SC_{dur} + W_{F0} * SC_{F0}$

接続コスト
$W_{env} + W_{F0c} + W_{spg} = 1$　　（重み）
$CC = W_{env} * SC_{env} + W_{F0c} * SC_{F0c} + W_{spg} * SC_{spg}$

素片候補系列内ターゲットコスト
$sumtc = pow(TC_1, tarp) + pow(TC_2, tarp) + \cdots + pow(TC_n, tarp)$
素片候補系列内接続コスト
$sumcc = pow(CC_1, conp) + pow(CC_2, conp) + \cdots + pow(CC_n, conp)$

統合コスト
$W_{tar} + W_{con} = 1$　　（重み）
$IC = pow(sumtc, 1.0/tarp) * W_{tar} + pow(sumcc, 1.0/conp) * W_{con}$

また、サブコスト値はターゲットコストに対するノルム（素片候補系列内の音素数にて正規化）と接続コストに対するノルム（素片候補系列内の音節境界数にて正規化）の重み付け和として計算します。

SCcen の計算方法

HTS から出力されたメルケプストラム時系列データ（24 次、5msec シフト）について、各音素毎にその音素区間を 6 分割し、中心 2 区間分の平均を取り、まず 24 要素の配列を作成します。次に VQ phoneme centroid ファイル（vq_cent.txt.bin）で定義されるベクトルコードブックから、その配列に対応するベクトルを探索します。コードブックベクトル間の距離計算は距離テーブル（centcbdist.mat）の参照により行われますが、この際、平均スペクトル（メルケプストラム）の共分散行列対角成分（centphdcov.mat）を参照し、同種の音素における原点からの平均平方二乗距離によりベクトル間距離が正規化されます。この値を用いて、下記計算式よりサブコストが計算されます。

```
#define SQUARE(x) ((x) * (x))
```

dist=セントロイド間の正規化距離

SCcen = (2.537640 - 2.537640 * exp(-1.0 * SQUARE(dist/120.250303))) * 正規化係数

SCF0 の計算方法

F0 の違いに関するサブコスト関数は、ターゲットと素片候補の 5 分割された F0 軌跡の差を求め、それを平均した値を用いています。

x=ターゲット

y=素片候補

```
Sepnum=5;
for (k = 1; k < sepnum + 1; k++) {

  f0cost += MAX( 0.0F, (float)(-0.003039 + 2.666204 / (1.0 + exp(-29.547275 *
        ( abs( (x->data[k] - y->data[k]) / log(2.0) ) - 0.229315)))));
}
f0cost = f0cost / (float)sepnum * 正規化係数;
SCF0 = f0cost
```

SCdur の計算方法

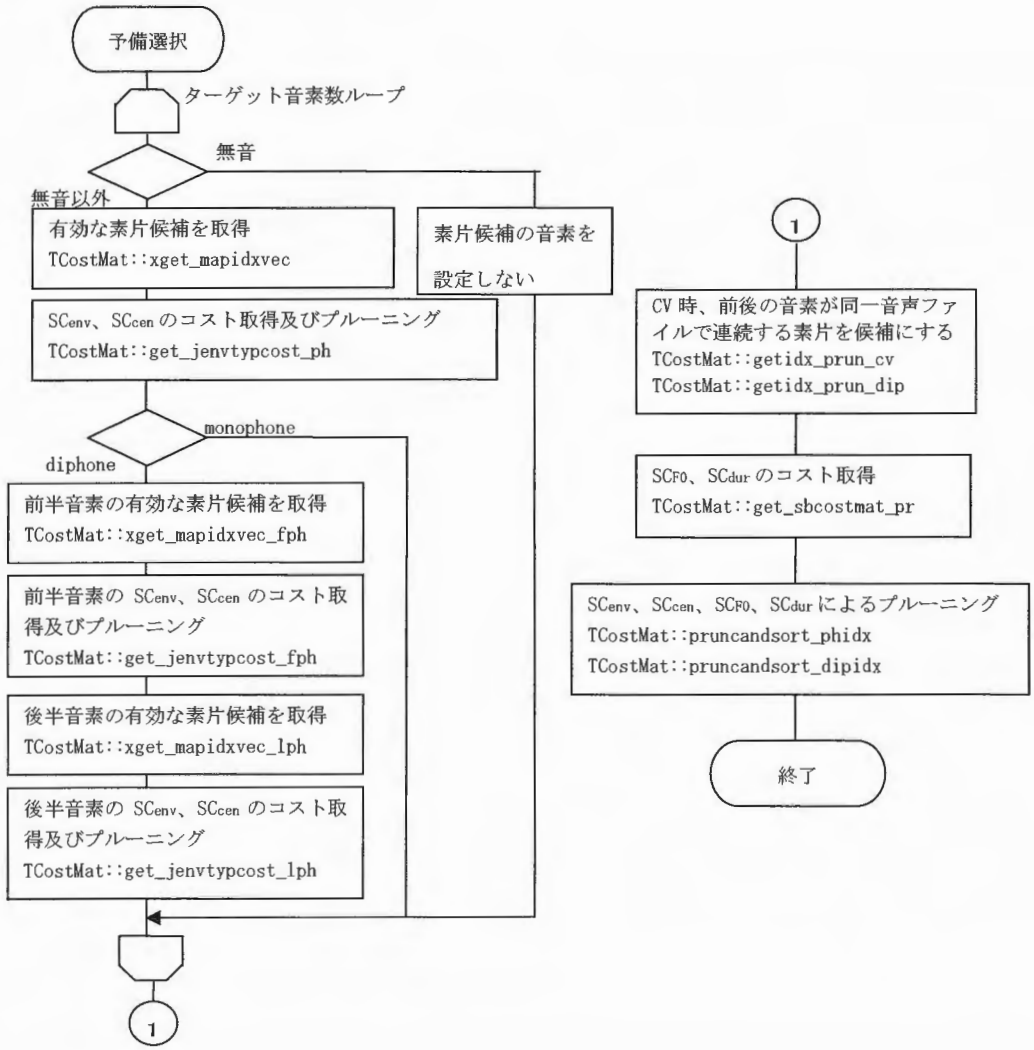
音素持続時間の違いに関するサブコスト関数は、ターゲットと素片候補間の音素持続長の差を音素長標準偏差により正規化した値を用いています。
音素長標準偏差は、データベースの素片候補統計量ファイル（statistic_rm-cl.txt）から取得します。

SCdur = 2.13255 * abs(ターゲット音素長 - 素片候補音素長) / 音素長標準偏差 * 正規化係数

6.6.5 予備選択

予備選択では計算回数を減らすため、多数の素片候補からプルーニング（絞込み）を行います。予備選択では、ターゲット情報の音素に対し、有効な素片候補をチェックしながら、SC_{cen}、SC_{env}のコスト計算を行い、各ターゲット音素の素片候補数がデータベースに1600以上（変更可能）存在する場合、プルーニングします。絞込みの条件は、各素片候補で求めたSC_{cen}、SC_{env}の合計が最小のものから予備選択コスト幅（デフォルト5.0）の範囲に収まることです。SC_{cen}、SC_{env}の予備選択後、SC_{dur}、SC_{F0}のコスト計算を行い、さらに予備選択を行います。プルーニングされた各音素の候補数が100より多い場合、SC_{cen}、SC_{env}、SC_{dur}、SC_{F0}の合計を小さいものから順に100位までを素片候補とします。（100位以降で100位と同一のコストは対象内として扱います）

予備選択処理フロー



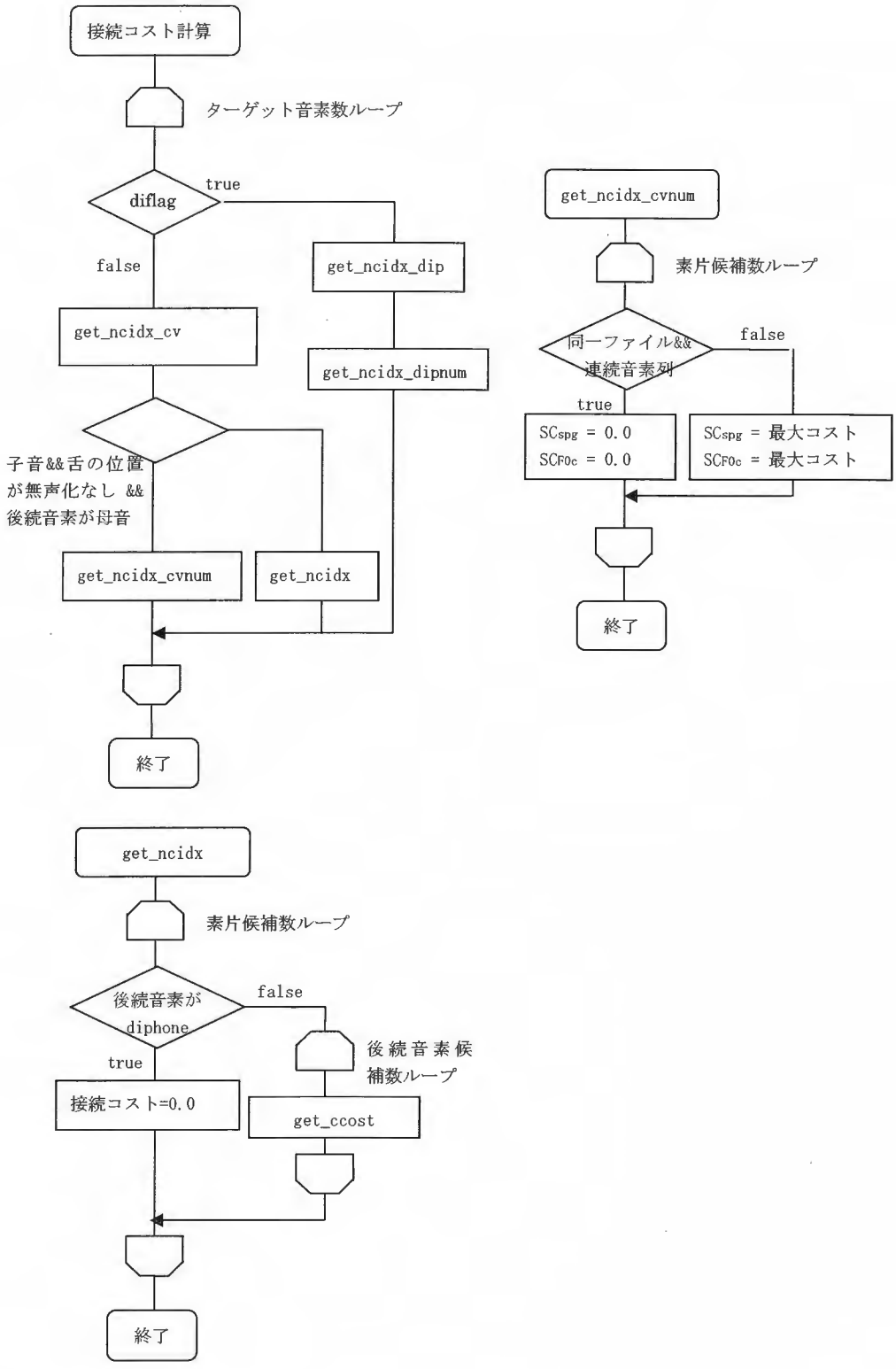
６．６．６ 接続コストの計算

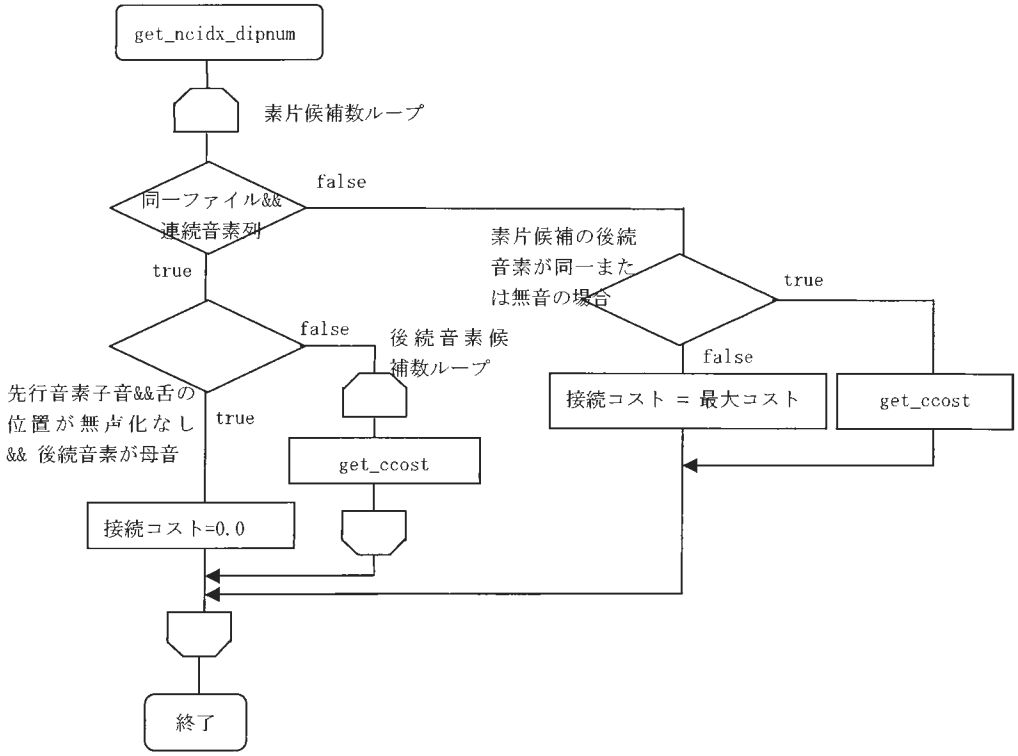
予備選択でプルーニングされた素片候補に対し、Idxs クラスで、SC_{F0c}、SC_{spg} のサブコスト計算を行い、接続コストを求めます。

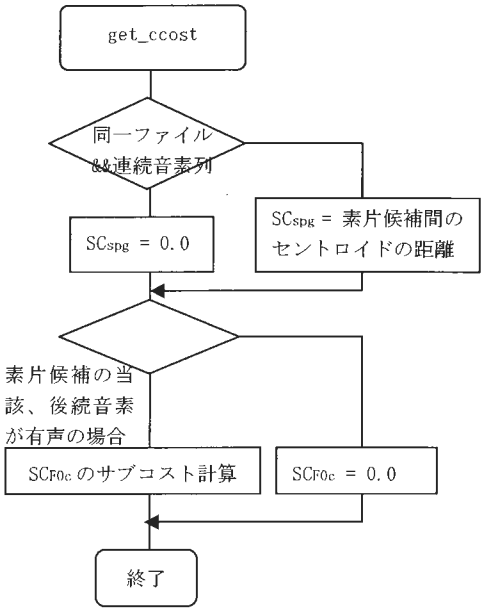
SC_{spg} は、vq_phb.txt.bin（VQ around phoneme boundary ファイル）、vq_vc.txt.bin（VQ around vowel center ファイル）、cbdist.mat（Centroid-Distance ファイル）を用い、素片候補間境界前後のフレーム間のケプストラム距離よりサブコストを求めます。

SC_{F0c} は、vq_phb.txt.bin（VQ around phoneme boundary ファイル）を用い、サブコストを求めます。
X = 当該素片候補
y = 後続素片候補
SCF0c = (-0.027277 + 1.507379 / (1.0 + exp(-25.878727 * (素片候補間境界付近の logF0 の違い / log(2.0) - 0.154329)))) * 正規化係数

SC_{env} は、聴覚実験から得た音素環境代替コストテーブルより求めます。先行音素のコスト値は、HTS で予測された音素列の先行音素、当該素片候補の先行音素をもとに、実験データである先行音素の音素環境代替テーブルの値から取得します。
後続音素の音素環境代替コストも同様にして値を取得します。
取得した値を次式のように計算します。（予備選択時計算）
SC_{env} =（(先行音素の環境代替コスト値＋後続音素の環境代替コスト値) / 2.0) * 正規化係数



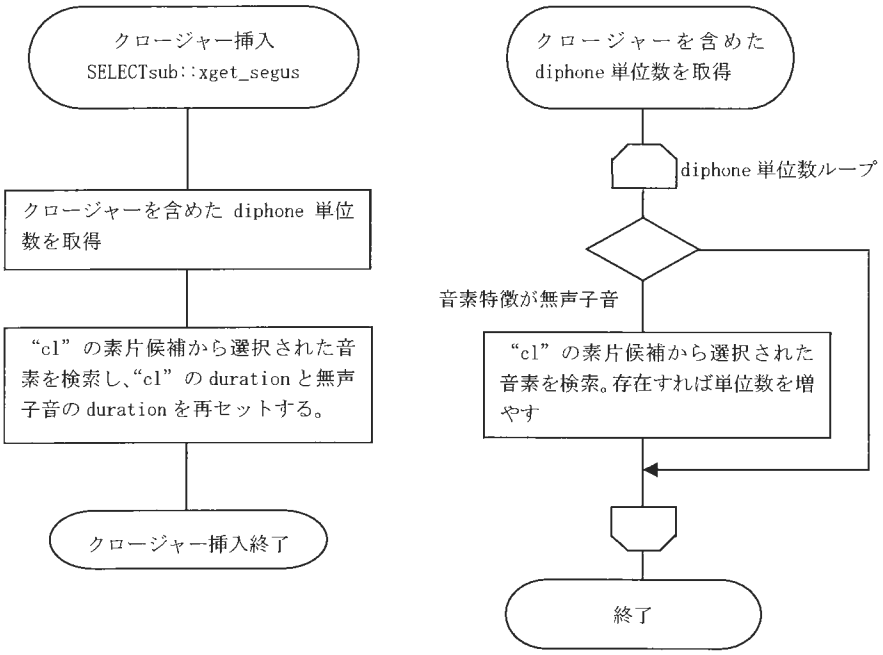




6.6.7 クロージャーの挿入

素片選択終了後、クロージャーの挿入を行います。素片選択された音素が無声子音の場合、データベースを検索し、素片選択された音素の先行音素に”cl”が含まれるかチェックを行います。含まれている場合、クロージャーを挿入します。

クロージャー挿入処理フロー

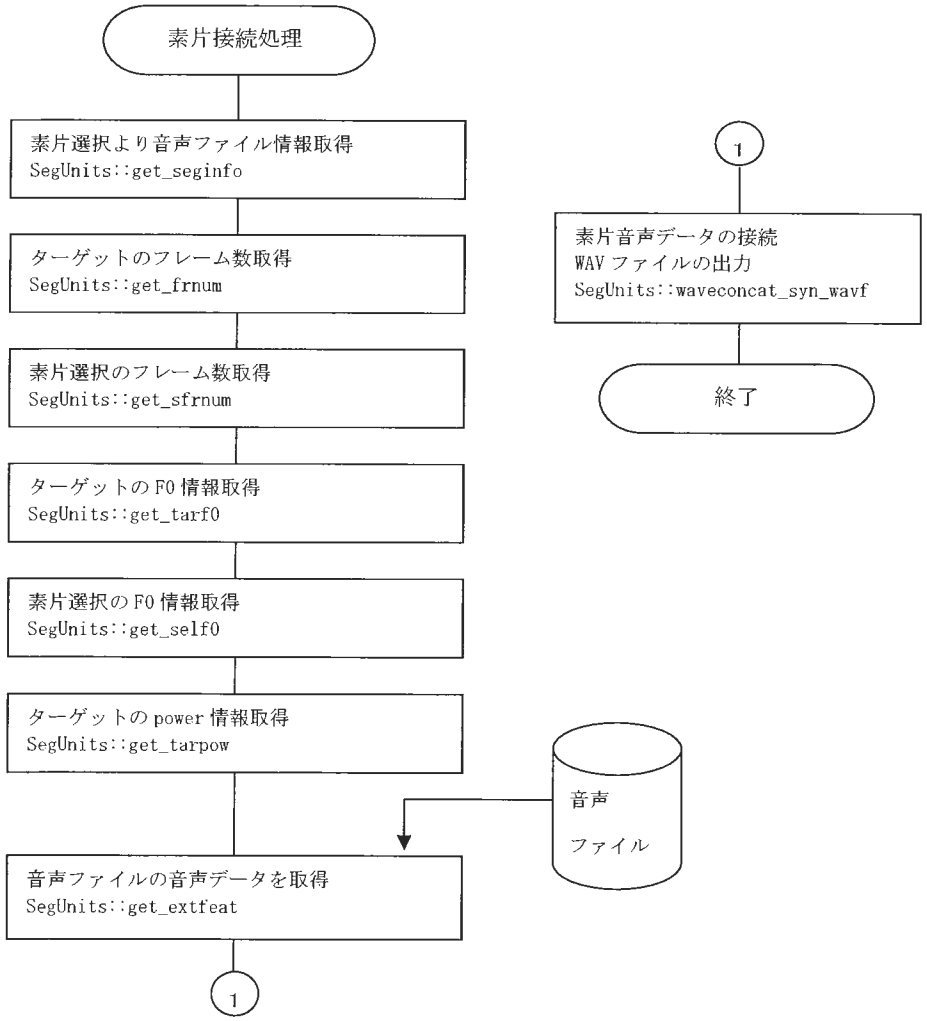


６．７ 素片接続処理

素片接続処理では、素片選択情報より各素片データを音声ファイルから取得し、素片間の接続を行います。

SegUnits クラスのコンストラクタは、音声ファイルから音声データを取得します。各素片の音声データは、取得する音素長分のデータに前後１フレーム（5ms)分を加えたデータ（のりしろ）を取得します。接続処理では、のりしろ部分で接続ポイントを検索し、そのポイントで窓掛けを行い、オーバーラップすることで接続をスムーズにします。

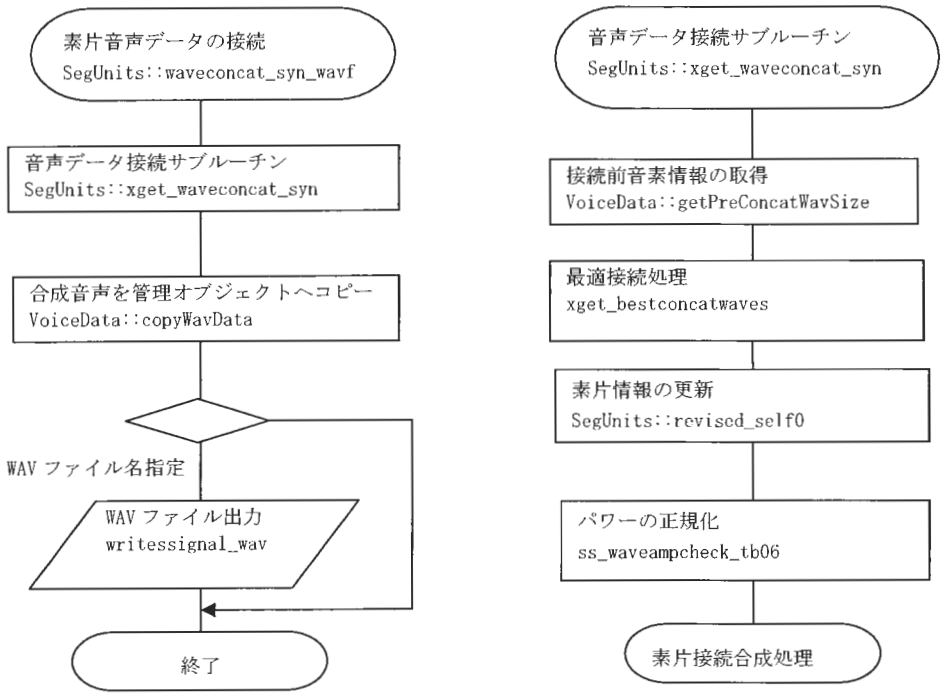
素片接続処理フロー

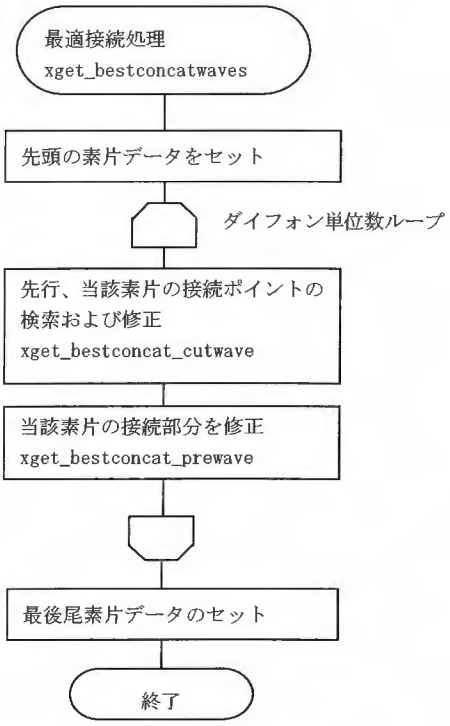


最適接続処理は、先行素片、当該素片ののりしろ部分を比較し、接続ポイントを検索し、修正を行います。
xget_bestconcat_cutwave 関数：接続ポイントの検索および先行素片の修正。
xget_bestconcat_prewave 関数：当該素片の接続部分の修正。

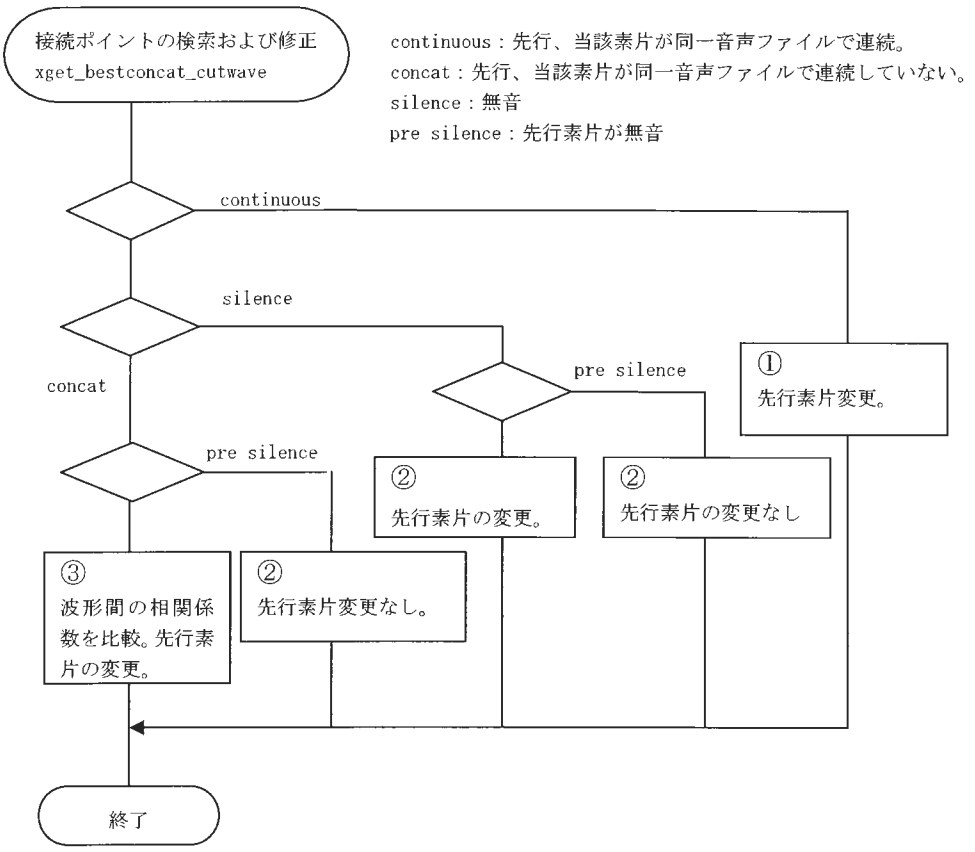
修正された音声データは、double から short への変換時にオーバーフローが発生しないようパワーの正規化（ss_waveampcheck_tb06 関数）を行い、合成結果を管理するオブジェクト（VoiceData クラス）へデータをコピーします。ファイル名が指定された場合、WAV ファイルを出力します。

素片音声データの接続の処理フロー





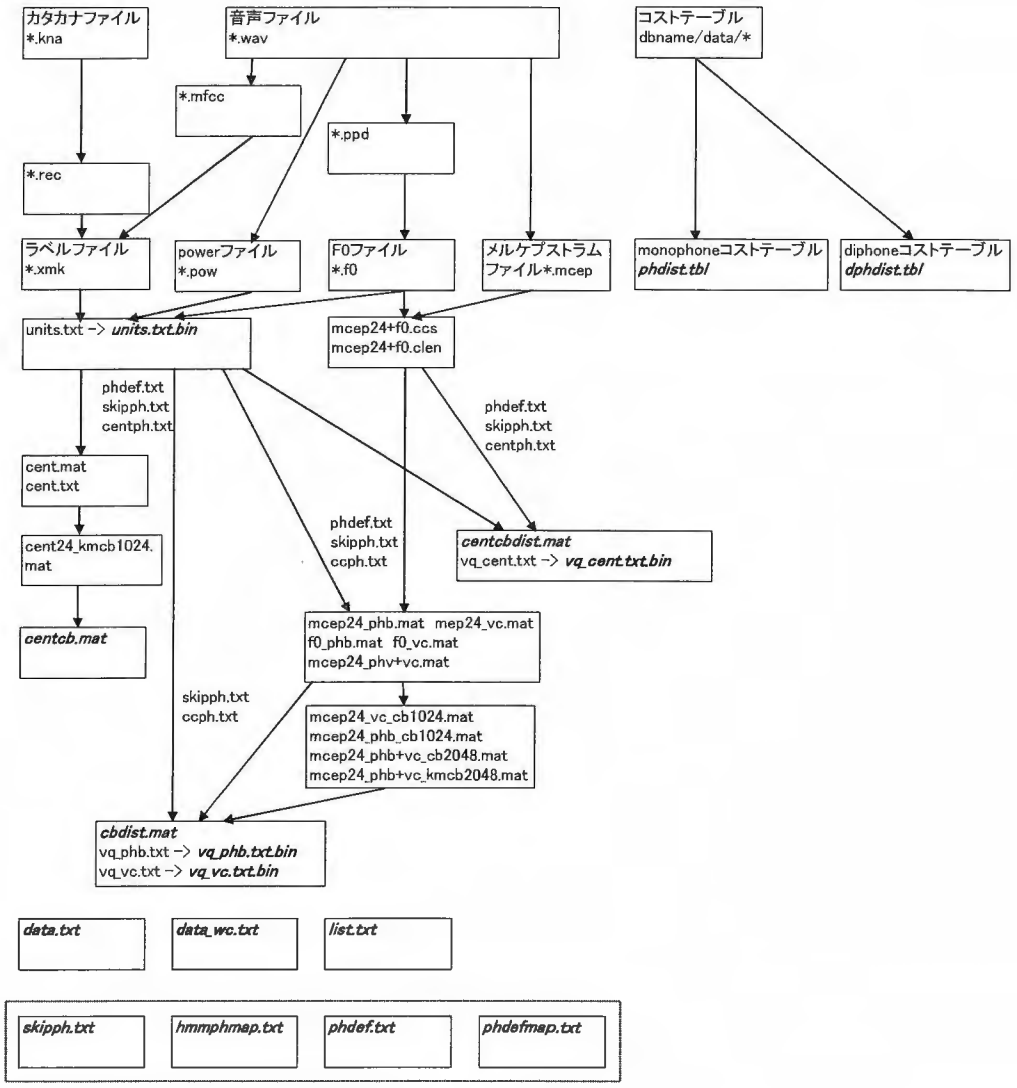
①先行、当該の素片が、同一音声ファイルかつ並びが連続している場合、のりしろの幅で先行素片の接続部分に窓掛けを行い、オーバーラップします。(音声ファイルから音声データ取得する際に、ターゲットと音声ファイル間のパワー調整を行っているため、ここでギャップを解消します。)
②先行、当該の素片のどちらかが無音の場合、音声データの値の差が最も小さいポイントを接続ポイントとし接続します。
③先行、当該の素片が、音声ファイルで連続していない場合(①以外の場合)、先行素片と当該素片の低域位相をそろえるために波形間の相関係数が高いポイントで、先行および当該素片の接続部分に窓掛けを行い、オーバーラップします。



7．データベース作成（AutoDBmake）

AutoDBmake ツールは XIMERA エンジンで使用するデータベースを作成するツールです。

7．1 AutoDBmake の流れ



*斜体太字で表されているファイルは XIMERA 合成エンジンのロードに必要なファイルです。

7．2 AutoDBmake ファイル一覧

以下は、XIMERA 合成エンジンのロードに必要なデータベースのファイル一覧です。

ファイル名	内容
cbdist.mat	音響的特徴コードブックのベクトル間距離テーブル
centcb.mat	ケプストラムセントロイドベクトルのコードブック
centcbdist.mat	ケプストラムセントロイドコードブックのベクトル間距離テーブル
centphdcov.mat	各音素の平均スペクトルの共分散行列対角成分ファイル
data.txt	素片選択用データベースの情報
data_wv.txt	素片接続で参照するコーパスの情報
dphdist.tbl	音素環境代替コストテーブル（diphone）
hmmphmap.txt	HTS 出力音素を素片選択で使用する音素片へ変換するルール（HMM 代替音素テーブル）
list.txt	データベースに含まれる Wave ファイルリスト
phdef.txt	音素定義ファイル
phdefmap.txt	コーパス中にほとんど存在しない音素を他の音素へ変換する規則（コーパス代替音素テーブル）
phdist.tbl	音素環境代替コストテーブル（monophone）
skipph.txt	スキップ音素リスト
statistic_rm-cl.txt	素片候補統計量ファイル
units.txt.bin	units（素片情報）ファイル
vq_cent.txt.bin	ケプストラムセントロイド情報（バイナリファイル）
vq_phb.txt.bin	音素境界における音響的特徴（バイナリファイル）
vq_vc.txt.bin	母音中心における音響的特徴（バイナリファイル）

7．3 出力ファイルフォーマット

AutoDBmake ツールにより出力されるファイルのフォーマットについて説明します。

7．3．1 ラベルファイル(*.xmk、*.mk)

音声ファイル中の各音素のスタートタイムと音素ラベルを表します。

AutoDBmake ツールによる Auto Align (align ボタン) の結果は、拡張子が xmk になります。拡張子が mk のファイルは Auto Align の結果に手修正を行ったファイルです。

スタートタイム(sec)	音素
0.000000	sil
1.008000	a
1.114410	r
1.134000	a
1.242970	y
1.344000	u
:	
4.419080	d
4.434000	a
4.542000	sil
5.482375	__END__

*__END__は音声データの終わりを表します。

7．3．2 ピッチファイル(*.xf0、*.f0)

音声ファイルの基本周波数（Hz）を10ms毎に記録します。

AutoDBmakeツールによるAuto F0（f0ボタン）の結果は、拡張子がxf0になります。拡張子がf0のファイルはAuto F0の結果に手修正を行ったファイルです。

ピッチの高さ
0.000000
0.000000
0.000000
0.000000
:
187.068546
193.739284
203.397436
209.745142
:
0.000000
0.000000
0.000000
0.000000

7．3．3 パワー(*.pow)

音声ファイルのパワーを 10ms 毎に記録します。
(バイナリファイル)

パワー［音声ファイル（ms）/10ms］個
power1 power2 power3･power4･･･

7．3．4 メルケプストラムファイル(*.mcep)

[音声ファイル長（ms）/フレームシフト（5ms）][24]の形式で24次メルケプストラムのうち0次の係数を除く値を記録します。
(バイナリファイル)

例)（テキスト表示）

	0	1	・・	22	23
0	mcep(0, 1)	mcep(0, 2)	・・	mcep(0, 23)	mcep(0, 24)
1	mcep(1, 1)	mcep(1, 2)	・・	mcep(1, 23)	mcep(1, 24)
L/5-1	mcep(L/5-1, 1)	mcep(L/5-1, 2)	・・	mcep(L/5-1, 23)	mcep(L/5-1, 24)
L/5	mcep(L/5, 1)	mcep(L/5, 2)	・・	mcep(L/5, 23)	mcep(L/5, 24)

*音声ファイル長（ms）＝ L

7．3．5 音素定義ファイル(phdef.txt)

音素名およびその音素の素性を定義します。

na　：音素名
cat　：分類：s = 無音　v = 母音　c = 子音　e = その他
uv　：有声無声：+ = 有声　- = 無声
slt　：無音の形式：d = 促音　p = 文中　s = 文頭，文末　c = 閉鎖区間
　　0 = 母音，子音，その他
lng　：母音の長さ：+ = 長母音　- = 短母音　0 = 母音以外
tp　：舌位置：1 = 前　2 = 中間　3 = 後ろ　- = 母音以外
op　：開口度：1 = 広い　2 = 中間　3 = 狭い　- = 母音以外
typ　：子音の調音様式：s = 半母音　n = 鼻音　l = 流音　p = 破裂音
　　f = 摩擦音　a = 破擦音
　　0 = 子音以外
plc　：調音位置：b = 唇　d = 歯　p = 口蓋　v = 軟口蓋　u = 口蓋垂
　　g = 声門　0 = 子音以外
ct　：拗音化：+ = 拗音あり　- = 拗音なし
db　：促音付：+ = 促音あり　- = 促音なし
tpt　：無声化の際の舌位置：1 = 前　2 = 中間　3 = 後ろ　- = 無声化なし
opt　：無声化の際の開口度：1 = 広い　2 = 中間　3 = 狭い　- = 無声化なし

na	cat	uv	slt	lng	tp	op	typ	plc	ct	db	tpt	opt
a	v	+	0	-	2	1	0	0	-	-	-	-
i	v	+	0	-	1	3	0	0	-	-	-	-
u	v	+	0	-	3	3	0	0	-	-	-	-
e	v	+	0	-	1	2	0	0	-	-	-	-
o	v	+	0	-	3	2	0	0	-	-	-	-
aa	v	+	0	+	2	1	0	0	-	-	-	-
:	:	:	:	:	:	:	:	:	:	:	:	:
>	e	-	0	0	-	-	0	0	-	-	-	-

７．３．６ コーパス代替音素マッピングファイル(phdefmap.txt)

コーパス中のサンプルが少ない音素を他の音素へ変換する規則を記述しています。

データベース中の変換対象音素セグメントが指定個数に満たない場合、合成に使用される音素をターゲット音素に変更します。

例)

変換対象音素	ターゲット音素	音素数
aa	a	50
fU	f	50

７．３．７ HMM 代替音素変換テーブル（hmmphmap.txt）

HTS 出力音素を素片選択で使用する音素片へ変換するルール（HMM 代替音素テーブル）です。

HMM 合成出力音素	XIMERA 使用音素
a a	aa
Q sh	Qsh

７．３．８ スキップ音素リスト(skipph.txt)

データベース作成時の特徴量算出から除外するための音素リストです。また合成エンジンでは、ターゲット音素作成時にスキップ音素がある場合、先行、後続音素の置き換えに利用しています。

```
Q  cl  >
```

7．3．9 音素環境代替コストテーブル（monophone）（phdist.tbl）

音素環境代替コストテーブル（data/tbl/phdist/以下のテーブル）です。
（バイナリファイル）

例）（テキスト表示）

a										
pre										
N	a	e	i	o	‥	‥	w	y	z	sil
2.2	2.3	2.3	4	3.9	‥	‥	50	50	50	50
4.3	3.1	4.3	4.7	3.9	‥	‥	50	50	50	50
2	3.5	0.4	0.7	3.9	‥	‥	50	50	50	50
2.3	4.3	1.9	0.4	4.3	‥	‥	50	50	50	50
3.9	1.6	4.3	4.7	0.4	‥	‥	50	50	50	50
2.4	3.1	3.5	3.4	2	‥	‥	-1	-1	-1	-1
-1	-1	-1	-1	-1	‥	‥	-1	-1	-1	-1
-1	-1	-1	-1	-1	‥	‥	-1	-1	-1	-1
-1	-1	-1	-1	-1	‥	‥	-1	-1	0	-1
-1	-1	-1	-1	-1	‥	‥	-1	-1	-1	0
next										
N	a	e	i	o	‥	‥	w	y	z	sil
1.3	2.7	1.3	2.7	1.3	‥	‥	50	50	50	50
2.7	3.1	3.5	2.7	2.7	‥	‥	50	50	50	50

*上記例は、[a]（対象音素）に対して[pre]以下は先行音素に対するコストテーブル、[next]以下は後続音素に対するコストテーブルです。横軸は抽出環境における先行または、後続音素で、縦軸は使用環境における先行または、後続音素です。

７．３．１０ 音素環境代替コストテーブル（diphone）（dphdist.tbl）

ダイフォン単位の音素環境代替コストテーブル（data/tbl/dphdist/以下のテーブル）です。

（バイナリファイル）

例）　（テキスト表示）

a										
pre										
N	a	e	i	o	･･	･･	w	y	z	sil
1.5	2.3	2.6	2.6	4	･･	･･	50	50	50	50
4.3	3.1	4.3	4.7	3.9	･･	･･	50	50	50	50
0.6	3.5	1	2.3	2.8	･･	･･	50	50	50	50
1.9	4.3	2.1	0.5	1.9	･･	･･	50	50	50	50
2.7	1.6	2.1	2.5	1.1	･･	･･	50	50	50	50
2	3.1	1.1	1.2	2.5	･･	･･	-1	-1	-1	-1
50	50	50	50	50	･･	･･	-1	-1	-1	-1
50	50	50	50	50	･･	･･	-1	-1	-1	-1
50	50	50	50	50	･･	･･	2.6	2.6	0.9	50
2.3	2.3	2.3	2.3	2.3	･･	･･	2.3	2.3	2.3	0
next										
N	a	e	i	o	･･	･･	w	y	z	sil
1.5	2.3	2.6	2.6	4	･･	･･	50	50	50	50
4.3	3.1	4.3	4.7	3.9	･･	･･	50	50	50	50

*上記例は、[a]（対象音素）に対して[pre]以下は先行音素に対するコストテーブル、[next]以下は後続音素に対するコストテーブルです。横軸は抽出環境における先行または、後続音素で、縦軸は使用環境における先行または、後続音素です。

7．3．11 リストファイル(list.txt)

合成に使われる音声ファイルのリストです。

例)

ファイル名
A/F009_ATR503_A01_T01
A/F009_ATR503_A02_T01
A/F009_ATR503_A03_T01
A/F009_ATR503_A04_T01
A/F009_ATR503_A05_T01
A/F009_ATR503_A06_T01
:

７．３．１２ 素片候補統計量ファイル(statistic_rm-cl.txt)

データベース中の各音素の素片情報の統計が格納されています。

フィールド１： 音素
フィールド２： 平均：音素長
フィールド３： 標準偏差：音素長
フィールド４： 平均：log F0
フィールド５： 標準偏差：log F0
フィールド６： 平均：log10(Power)
フィールド７： 標準偏差：log10(Power)
フィールド８： データベース中の音素数

例)

フィールド１	フィールド２	フィールド３	フィールド４	フィールド５	フィールド６	フィールド７	フィールド８
a	86.986	24.316	5.493	0.236	6.820	0.538	3503
i	72.849	28.322	5.538	0.230	6.671	0.497	2305
u	61.606	30.813	5.518	0.255	6.435	0.549	1722
e	85.078	26.328	5.514	0.236	6.813	0.346	1741
:	:	:	:	:	:	:	:

７．３．１３ 素片情報ファイル（units.txt.bin）

データベース中の各音素のターゲット情報を記述しています。
（バイナリファイル）

例）（テキスト表示）

```
# 1-st column: Phoneme
# 2-nd column: Start Time [ms]
# 3-rd column: Duration [ms]
# 4-th column: Duration [ms]
# 5->9-th column: average of log F0 in each devided part
# 10->12-th column: average of log10(Power) in each devided part
# 13-th column: Phoneme-Number
# 14-th column: File Index
# 15-th column: Tone
sil 0 1008 1008.000 0.000 0.000 0.000 0.000 0.000 0.410 0.459 0.548 0 0 -1
a 1008 106 106.410 5.231 5.291 5.360 5.442 5.513 6.140 6.984 6.781 1 0 -1
r 1114 20 19.590 5.527 5.574 5.574 5.710 5.710 6.326 6.008 6.617 2 0 -1
```

7．3．14 ケプストラムセントロイドベクトルのコードブック(centcb.mat)

ケプストラムセントロイドベクトルのコードブックを表します。

24次MFCCのうち0次の係数を除く要素24個のベクトルの系列が、インデックス0のベクトルから順に格納されています。
(バイナリファイル)

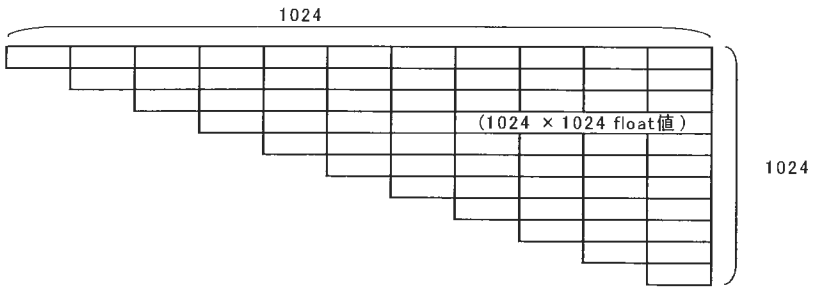
(テキスト表示)

	1	2	3	・・	22	23	24
0				・・			
1	(24 × 1024 float値)			・・			
1022				・・			
1023				・・			

７．３．１５ ケプストラムセントロイドコードブックに含まれるベクトル間の距離テーブル(centcbdist.mat)

ケプストラムセントロイドコードブックに含まれる各ベクトル間の距離を表します。(バイナリファイル)

centcb.mat に含まれる各ベクトル間の距離を事前に計算したものが格納されています。対称行列なので、ファイルには上三角の要素のみ以下の順序で格納されています。(テキスト表示)



7．3．16 ケプストラムセントロイド情報(vq_cent.txt.bin)

data/centph.txt 中の音素を対象に、素片中央部における MFCC 値に対応するコードブックのインデックスが格納されています。
(バイナリファイル)

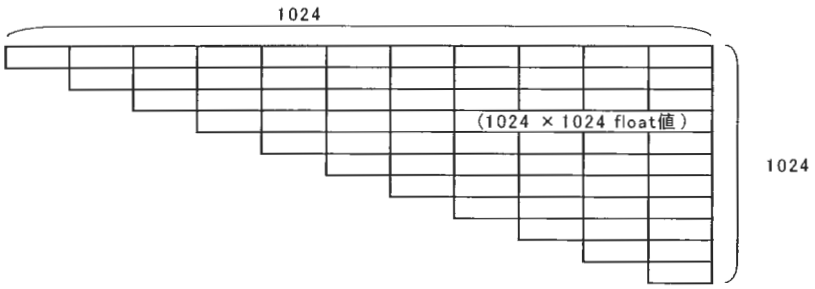
例)（テキスト表示）

ファイル Index	ファイル中の音素 Index	コードブックのインデックス
0	1	845
0	2	77
0	3	833
:	:	:

７．３．１７ 音響的特長コードブックに含まれるベクトル間の距離テーブル(cbdist.mat)

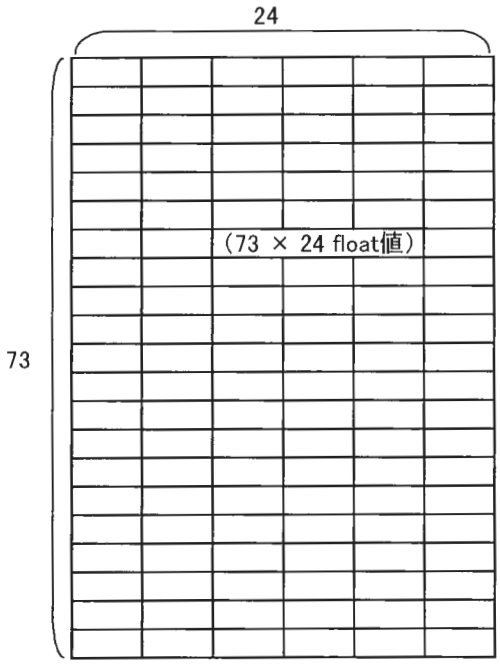
MFCC不連続に関するサブコスト計算のための素片境界MFCCコードブックに含まれる各ベクトル間の距離のテーブルです。
（バイナリファイル）

対称行列なので、ファイルには上三角の要素のみ以下の順序で格納されています。（テキスト表示）



７．３．１８ 各音素の平均スペクトルの共分散行列対角成分ファイル(centphdcov.mat)

各音素の平均スペクトルの共分散行列対角成分が[音素数（73）×24]の配列で格納されています。
(バイナリファイル)



７．３．１９ 音素境界周辺の音響的特徴情報(vq_phb.txt.bin)

素片境界付近の音響的特徴に対応する音響的特徴コードブックのインデックスが格納されています。
（バイナリファイル）

フィールド１： list.txt のファイル No.　units.txt の 14-th column に対応
フィールド２： ファイル中の音素の No.　units.txt の 13-th column に対応
フィールド３： コードブックのインデックス
フィールド４： コードブックのインデックス
フィールド５： コードブックのインデックス
フィールド６： コードブックのインデックス
フィールド７： log F0

例）（テキスト表示）

フィールド１	フィールド２	フィールド３	フィールド４	フィールド５	フィールド６	フィールド７
0	0	0	0	380	1574	0.000000
0	1	416	1090	1090	1090	5.526741
0	2	1090	1090	764	636	5.710052
0	3	732	732	1210	604	5.896154
:	:	:	:	:	:	:

７．３．２０ 母音中心周辺の音響的特徴情報(vq_vc.txt.bin)

data/ccph.txt 中の音素を対象に、素片中心付近の音響的特徴について、対応する音響的特徴コードブックのインデックスが格納されています。
（バイナリファイル）

フィールド１： list.txt のファイル No.　units.txt の 16-th column に対応
フィールド２： ファイル中の音素の No.　units.txt の 15-th column に対応
フィールド３： コードブックのインデックス
フィールド４： コードブックのインデックス
フィールド５： コードブックのインデックス
フィールド６： コードブックのインデックス
フィールド７： log F0

例）（テキスト表示）

フィールド1	フィールド2	フィールド3	フィールド4	フィールド5	フィールド6	フィールド7
0	1	1062	1606	892	1318	5.373965
0	3	732	972	972	76	5.798676
0	5	1418	668	1386	1130	5.873682
0	7	604	1834	348	1082	5.728011
:	:	:	:	:	:	:

7.3.21 データベースファイルリストおよびパラメータ(data.txt)

素片選択のパラメータおよび参照するファイルとそのパスを記入します。

エンジン参照タグ	参照ファイルおよび数値
-phdeff	音素定義ファイル
-skipphf	スキップ音素リストファイル
-phdistf	monophone 代替コストテーブル
-dphdistf	diphone 代替コストテーブル
-unitsf	units (素片情報) ファイル
-listf	データベースに含まれる音声ファイルリスト
-cbdistf	音響的特徴コードブックのベクトル間距離テーブル
-vqiff	素片境界付近の音響的特徴
-cvqiff	素片中心付近の音響的特徴
-centiff	素片中心付近のセントロイドテーブル
-centcbf	ケプストラムセントロイドベクトルのコードブック
-centcbdistf	ケプストラムセントロイドコードブックのベクトル間の距離テーブル
-shift	フレームシフト[単位 ms] (10)
-sepnum	フレームに対する F0 の個数 (5)
-phmapf	代替音素マッピングファイル
-hmmdim	メルケプストラムの要素数 (24)
-hmmshift	HMM シフト (5)
-hmmphmapf	HTS 出力音素を素片選択で使用する音素片へ変換するルール (HMM 代替音素テーブル)
-centphdconvf	各音素の平均スペクトルの共分散行列対角成分ファイル
-stf	素片候補統計量ファイル

*( )内の数値は AutoDBmake ツールでデータベースを作成した場合のデフォルト値です。AutoDBmake ツールの GUI 上からはこれらのパラメータの変更が不可能となっています。デフォルト値以外でデータベースを作成する場合はソースを参照の上、必要箇所を変更してください。

7．3．22 コーパス情報およびパラメータファイル(data_wvc.txt)

素片接続に使用する音声ファイルのパスや合成に使用するパラメータおよび、音声ファイルの拡張子が記入されています。

エンジン参照タグ	参照ファイルおよび数値
-f	サンプリングレート[単位：Hz]
-shift	のりしろの幅[単位：ms]（デフォルト値：5.0ms）
-f0shift	データベースのF0フレームシフト(5)
-wavdir	データベースに含まれる音声ファイルのパス
-wavsign	ファイル拡張子（wav）
-wavcacheidx	頻度情報で使用する音声ファイル名一覧
-wavcacheseg	頻度情報ファイル
-wavcachefreq	頻度情報作成に用いたテストデータ合成結果の音素数に対する使用頻度[単位：%]（デフォルト値：20.0）

合成時の高速化に関するオプションについて

合成時、音声ファイルのopen、readに時間がかかるため、データベース読込時、頻度情報ファイル（-wavcachesegで指定されたファイル）を元に先読みを行い、OSのディスクキャッシュに読み込ませます。-wavcachefreqは先読みするデータ量を指定します（値が小さいほど大量に先読みを行います）。これにより、素片接続処理が高速化されます。ただし、データベースの読み込みには非常に時間がかかるため、使用には注意が必要です。頻度情報ファイルの作成方法はXIMERA GUIユーザーズマニュアルを参照してください。

設定例：

```
-wavcacheidx fidx_50k.txt
-wavcacheseg freq_50k.txt
-wavcachefreq 20.0
```

7．4 その他のファイル

その他、[データベース名]/data/フォルダ以下にあるデータベース作成時に参照するファイルやコストテーブルの一覧を以下に示します。

ファイル名	内容
ccph.txt	音素中心接続を考慮する音素のリスト
centph.txt	音素中心付近における平均スペクトルを計算する音素のリスト
next-cur-phmap.txt	mapping next-current-phoneme ファイル
next-phmap.txt	mapping next-phoneme ファイル
pre-cur-phmap.txt	mapping pre-current-phoneme ファイル
pre-phmap.txt	mapping pre-phoneme ファイル
tbl/phdist/data/*	これらのファイルは聴覚実験の結果から作成されたものです音素接続時の音素環境の代替によって生じる自然性劣化を表しています
next-cur-phmap_dip.txt	ダイフォン単位 mapping next-current-phoneme ファイル
next-phmap_dip.txt	ダイフォン単位 mapping next-phoneme ファイル
pre-cur-phmap_dip.txt	ダイフォン単位 mapping pre-current-phoneme ファイル
pre-phmap_dip.txt	ダイフォン単位 mapping pre-phoneme ファイル
tbl/dphdist/data/*	これらのファイルは聴覚実験の結果から作成されたものですダイフォン単位接続時の音素環境の代替によって生じる自然性劣化を表しています

7．5 話者モデル

Auto Align が自動音素セグメンテーションを行う際に使用する音響モデル（HMM）および関連ファイルです。これらのファイルは ATR503 バランス文を元に作成されています。

Model ファイル	内容
biphone_F009_ATR503.mmf	F009　biphone HMM
monophone_F009_ATR503.alt	F009　代替音素リスト
monophone_F009_ATR503.mmf	F009　monophone HMM
monophone_F009_ATR503.offset	F009　オフセットデータ

7.6 プログラム処理内容

AutoDBmake の各プロセス毎に処理概要とファイルの入出力を説明します。

7.6.1 align

カナファイル(*.kna)からラベルファイル(*.xmk)を作成します。

プログラム名、入出力ファイルは以下のとおりです。

Mfcc.java

16kHz ヘッダー付音声ファイル（RIFF 形式）を読み込み、mfcc ファイルを作成します。外部コマンド HCopy を呼び出します。
16kHz 以外の Wave ファイルを使用した場合、外部コマンド srconv を使って 16kHz にダウンサンプリングした後、mfcc ファイルを作成します。srconv が対応可能なサンプリングレートは、8kHz、22.05kHz、32kHz、44.1kHz、48kHz です。ただし 16kHz、48kHz 以外の Wave ファイルに関してはサポート外とします。

（外部コマンド）
HCopy
srconv

input: [corpus]/*.wav

output: [corpus]/*.mfcc　　mfcc ファイル

Align.java

mfcc ファイルとカナファイルを元にラベルファイルを作成します。
Kana2Gra.java により HParse が読込み可能な文法ファイルリストを作成後、HVite により、HTK ラベルファイルを作成します。HTK ラベルファイルを元に *.xmk ファイルを作成します。

Kana2Gra.java

（外部コマンド）
HParse

HVite

input: [corpus]/*.kna
[corpus]/*.wav
[corpus]/*.mfcc

output: [corpus]/*.rec HTK ラベルファイル
[corpus]/*.xmk ラベルファイル

7.6.2 f0

F0の抽出

F0 取得の外部コマンド、pitch は Cambridge University, Chris Tuerk 作成の Pitch extractor and pitch period marking program を改良したものです。10ms 毎の F0 を抽出します。この際、Min Pitch と Max Pitch を、使用している音声の F0 の範囲に近い値にする事により、より精度の高い結果が得られます。

プログラム名、入出力ファイルは以下のとおりです。

Pitch.java
(外部コマンド)
pitch

input:	[corpus]/*.wav	
output:	[corpus]/*.ppd	基本周期ファイル（F0 抽出のための入力ファイル）
	[corpus]/*.xf0	F0 ファイル

7．6．3 power

パワーの抽出

外部コマンドpowにより、10ms毎のパワーの値を取得します。

プログラム名、入出力ファイルは以下のとおりです。

Calpow.java

（外部コマンド）

pow

input: [corpus]/*.wav

output: [corpus]/*.pow　　パワーファイル

7．6．4 mcep

メルケプストラムの抽出
外部コマンド、straight-analysis により、[音声ファイル長（ms）/フレームシフト（5ms）][24]の24次メルケプストラム係数のうち0次の係数は除いた値を抽出します。

プログラム名、入出力ファイルは以下のとおりです。

Mcep.java
（外部コマンド）
straight-analysis

input: [corpus]/*.wav
[corpus]/*.xf0

output: [corpus]/*.mcep　　メルケプストラムファイル

7．6．5 phdtbl

音素環境代替コストテーブルを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_phdtbl.java
外部コマンド、mk_phdtbl により、聴覚実験の結果から決定された音素環境代替コストテーブルをバイナリファイルに書き換えます。

（外部コマンド）
mk_phdtbl

input:	phd_list.txt	読み込みファイルの一覧
	dphd_list.txt	読み込みファイルの一覧
	data/tbl/phdist/*	コストテーブル
	data/tbl/dphdist/*	コストテーブル
output:	data/tbl/phdist.tbl	monophone コストテーブル
	data/tbl/dphdist.tbl	diphone コストテーブル

7.6.6 units

unitsファイル、素片候補統計量ファイルの作成

外部コマンド、mk_units により、データベース中に含まれる各音素のターゲット情報、素片の統計量を phdef.txt、*.mk、*.f0、*.pow から作成します。以下にフォーマットを示します。

(units.txt.bin のフォーマット)

```
# 1-st column       : Phoneme
# 2-nd column       : Start Time [ms]
# 3-rd column       : Duration [ms]
# 4-th column       : Duration [ms]
# 5->9-th column: average of log F0 in each devided part
# 10->12-th column: average of log10(Power) in each devided part
# 13-th column: Phoneme-Number
# 14-th column      : File Index
# 15-th column      : Tone (未使用)
```

(statistic_rm-cl.txt)

```
# 1-st column       : Phoneme
# 2-nd column       : Average of Duration [ms]
# 3-rd column       : Standard Deviation of Duration
# 4-th column       : Average of log F0
# 5-th column       : Standard Deviation of log F0
# 6-th column       : Average of log10(Power)
# 7-th column       : Standard Deviation of log10(Power)
# 8-th column       : The number of phonemes
```

プログラム名、入出力ファイルは以下のとおりです。

MK_units.java

(外部コマンド)

mk_units

input: data/phdef.txt

[corpus]/*.xmk(*.mk)
[corpus]/*.xf0(*.f0)
[corpus]/*.pow

output: units.txt　　units ファイル
statistic.txt　　statistic ファイル

MK_stat.java
(外部コマンド)
mk_units

input: data/phdef.txt
[corpus]/*.xmk(*.mk)
[corpus]/*.xf0(*.f0)
[corpus]/*.pow

output: statistic_rm-cl.txt　　素片候補統計量ファイル

7．6．7 cost

.f0 ファイルと.mcep ファイルから音響特徴量ファイルを作成します。

外部コマンド、mk_ccost により、メルケプストラムファイルと F0 ファイルから音響特徴量ファイルを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_ccsf.java
　（外部コマンド）
　mk_ccost

　input:　[corpus]/*.xf0(*.f0)
　　[corpus]/*.mcep

　output: mcep24+f0.ccs　　concat cost ファイル
　　mcep24+f0.clen　　concat-cost length ファイル

7.6.8 centroid

ケプストラムセントロイド(mean vector)ファイルを作成します。

外部コマンド mk_cent により、centph.txt に含まれる音素のケプストラムセントロイドを取得します。

同時に concat-cost file リストと concat-cost length file リストを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_cent.java

(外部コマンド)

mk_cent

input: data/phdef.txt
data/skipph.txt
data/centph.txt
units.txt

output: cent.mat — centroid matrix ファイル
cent.txt — centroid information ファイル
mcep24+f0_ccs.list — concat-cost file リスト
mcep24+f0_clen.list — concat-cost length file リスト

７．６．９ vq-cbook

ケプストラムセントロイド(mean vector)のコードブックファイルを作成します。

外部コマンド vq-lbg と vq-kmean により、ケプストラムセントロイドベクトルのコードブックを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_centcbook.java

（外部コマンド）

vq-lbg

vq-kmean

input: cent.mat

output: centcb.mat　ケプストラムセントロイドのベクトルのコードブック

data/vq_cent/cent24_cb1024.mat　LBG アルゴリズムによるコードブック

data/vq_cent/cent24_kmcb1024.mat　k-means アルゴリズムによるコードブック

7.6.10 vq_cent

ケプストラムセントロイド情報ファイルと各音素の平均スペクトルの共分散行列対角成分ファイルを作成します。

外部コマンド mk_cent により、ケプストラムセントロイドコードブックのベクトル間距離テーブルを作成し、外部コマンド mk_phcov により各音素の平均スペクトルの共分散行列対角成分ファイルを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_vq_cent.java
(外部コマンド)
mk_cent

input: data/phdef.txt
data/skipph.txt
mcep24+f0_ccs.list
mcep24+f0_clen.list
data/centph.txt
units.txt

output: centcbdist.mat ケプストラムセントロイドコードブック
ベクトル間の距離テーブル
vq_cent.txt ケプストラムセントロイド情報ファイル

MK_phcov.java
(外部コマンド)
mk_phcov

input: data/phdef.txt
data/skipph.txt
mcep24+f0_ccs.list
mcep24+f0_clen.list
units.txt

output: centphdcov.mat 各音素の平均スペクトルの共分散行列対角

成分ファイル

centphmean.mat　　音素ごとの平均ベクトルファイル

7．6．11 ext_frm

音素境界、母音中心における音響的特徴(MFCC、F0) を抽出します。

外部コマンド ext_frm により、各音素の音素境界付近と音素中心接続を考慮する音素の中心付近における MFCC、log(F0)情報を作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_ext_frm.java
(外部コマンド)
ext_frm

input: data/phdef.txt
data/skipph.txt
mcep24+f0_ccs.list
mcep24+f0_clen.list
units.txt
data/ccph.txt

output:	mcep24_phb.mat	mcep phoneme boundary の Frame 抽出ファイル（cbook、VQ の入力ファイル）
	mcep24_vc.mat	mcep vowel center の Frame 抽出ファイル（cbook の入力ファイル）
	f0_phb.mat	f0 phoneme boundary の Frame 抽出ファイル（VQ の入力ファイル）
	data/f0_vc.mat	f0 vowel center の Frame 抽出ファイル(VQ の入力ファイル)
	mcep24_phb+vc.mat	Joined ファイル (cbook の入力ファイル)

7.6.12 cbook

音響的特徴コードブックの作成

外部コマンド vq_lbg と vq_kmean により、LGB アルゴリズム、k-means アルゴリズムを使ってコードブックを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_cbook.java

(外部コマンド)

vq-lbg

vq-kmean

input:	mcep24_phb.mat	
	mcep24_vc.mat	
	mcep24_phb+vc.mat	
output:	data/vq/mcep24_vc_cb1024.mat	LBG アルゴリズムによる Vowel center コードブック(Vector 1024)
	data/vq/mcep24_phb_cb1024.mat	LBG アルゴリズムによる Phonetic boundary コードブック(Vector 1024)
	data/vq/mcep24_phb+vc_cb2048.mat	LBG アルゴリズムによる Joined コードブック
	data/vq/mcep24_phb+vc_kmcb2048.mat	k-means アルゴリズムによる Joined コードブック(VQ の入力ファイル)

７．６．１３ vq

音響的特徴の VQ

外部コマンド mk_vqif により、音響的コードブックのベクトル間距離テーブルと、音素の中心付近および音素境界付近におけるコードブックのインデックスを抽出します。

プログラム名、入出力ファイルは以下のとおりです。

MK_vq.java
　(外部コマンド)
　mk_vqif

input: units.txt
　　data/skipph.txt
　　data/ccph.txt
　　mcep24_phb.mat
　　f0_phb.mat
　　data/vq/mcep24_phb+vc_kmcb2048.mat
　　mcep24_vc.mat
　　f0_vc.mat

output: cbdist.mat　　MFCC コードブック中のベクトル間距離テーブル (Centroid-Distance ファイル)
　　vq_phb.txt　　VQ around phoneme boundary label ファイル
　　vq_vc.txt　　VQ around vowel center label ファイル

7．6．14 mk_data

バイナリファイル、データファイルの作成

外部コマンドbin_unitsにより、units.txt、vq_phb.txt、vq_vc.txt、vq_cent.txtをバイナリファイルに変換し、XIMERA エンジンのデータベースファイルリストとパラメータを記録したファイルを作成します。

プログラム名、入出力ファイルは以下のとおりです。

MK_binfiles.java
(外部コマンド)
bin_units

input: data/phdef.txt
units.txt
vq_phb.txt
vq_vc.txt
vq_cent.txt

output: units.txt.bin
vq_phb.txt.bin
vq_vc.txt.bin
vq_cent.txt.bin

MK_data.java

output: data.txt
data_wvc.txt

7.6.15 Merge DB

複数のデータベースをマージし1つのデータベースにします。複数のデータベースをマージすることにより、中間ファイルサイズが 2GB を超えるような大規模なデータベースの作成も可能になります。

複数データベースより 1 つの units.txt ファイルを作成し、外部コマンド mk_cent によりセントロイドファイル（cent.mat）を作成し、その他、必要ファイルを作成した後、cont_vqinfo により、音素境界付近の VQ コードブック（vq_phb.txt）と、母音中心付近の VQ コードブック（vq_vc.txt）をマージします。セントロイドファイル、音素境界付近の VQ コードブック、母音中心付近の VQ コードブック作成以外は通常のデータベース作成と同様の動作をしますが、ラベルファイル作成、F0 抽出、パワー計算、メルケプストラム計算などはすでに作成されたファイルを使用します。

この場合、VQ コードブックの作成はせず、他の DB で作成した VQ コードブック（centcb.mat）ファイルをコピーして利用します。（ユーザーズマニュアル =データベース作成= 参照）

プログラム名、入出力ファイルは以下のとおりです。

```
MK_units.java
MergeDB.java     *1
MK_centcbook.java
MK_vq_cent.java
MergeDB.java     *2
MK_binfiles.java
MK_data.java

MergeDB.java      (*1)
    mk_cent
        input:  …/.xmk(*.mk)
                …/.xf0(*.f0)
                …/*.pow
                data/phdef.txt
                data/skipph.txt
                data/centph.txt
                mcep24+f0_ccs.list
                mcep24+f0_clen.list
```

```
                    units.txt

          output: cent.txt
                  cent.mat

MergeDB.java      (*2)
    cont_vqinfo.exe
        input:  vq_phb_txt.list
                vq_vc_txt.list
```

8．HTS トレーニング

拡張環境依存ラベルの作成、学習用特徴データファイルの作成、HMM ファイルの作成を行います。

＊HTK/HTS の詳細に関しては下記 URL をご参照ください。

HTS　http://hts.ics.nitech.ac.jp/

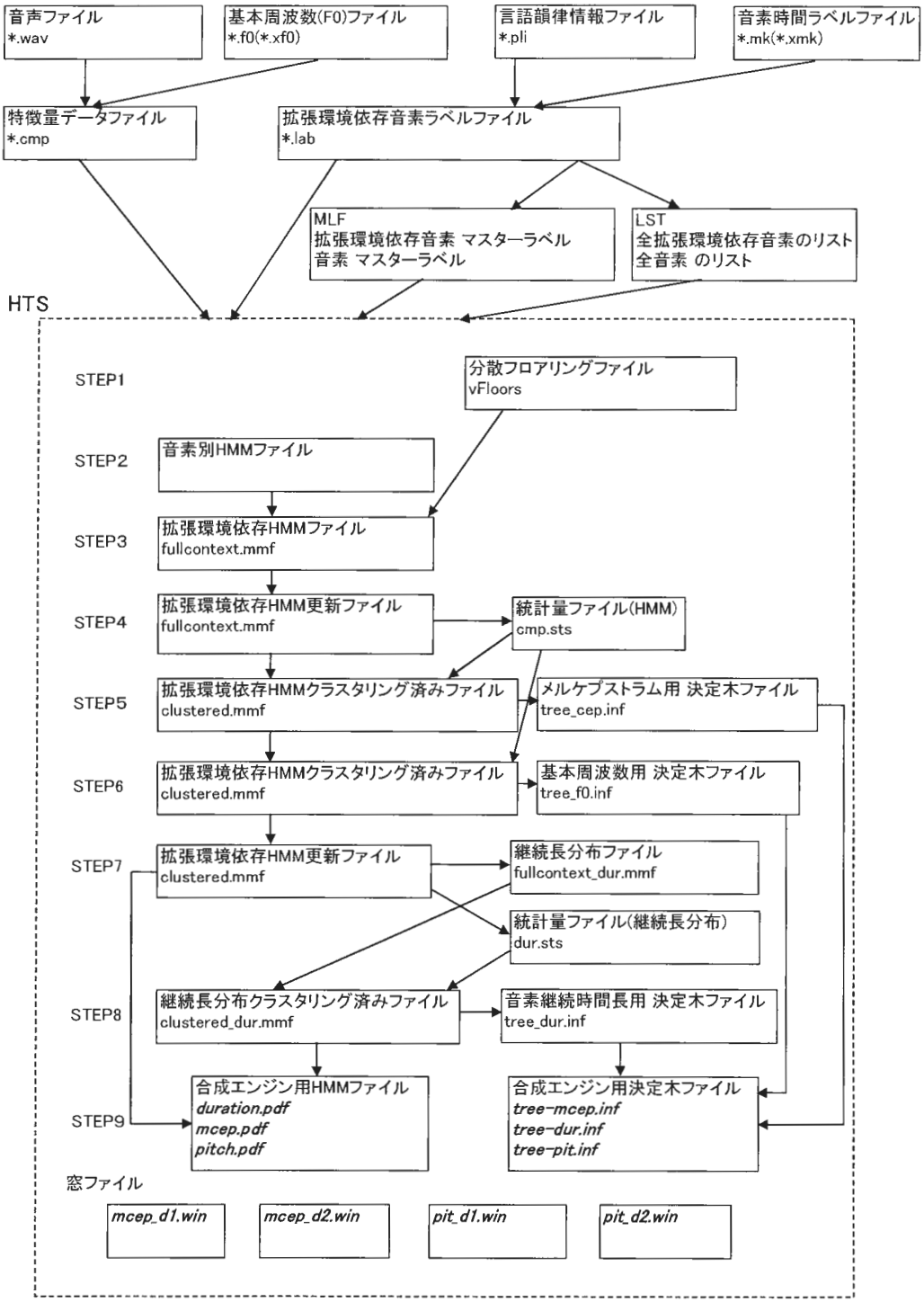
HTK　http://htk.eng.cam.ac.uk/

8．1 HTS Training ファイル一覧

XIMERA 合成エンジン用 HMM ファイル類

ファイル名	内容
duration.pdf	音素継続時間長用　HMM ファイル
mcep.pdf	メルケプストラム(スペクトル)用 HMM ファイル
pitch.pdf	基本周波数(ピッチ/F0)用　HMM ファイル
mcep_d1.win	メルケプストラム(スペクトル)用 窓ファイル(一次)
mcep_d2.win	メルケプストラム(スペクトル)用 窓ファイル(二次)
pit_d1.win	基本周波数(F0)用　窓ファイル(一次)
pit_d2.win	基本周波数(F0)用　窓ファイル(二次)
tree-dur.inf	音素継続時間長用　決定木ファイル
tree-mcep.inf	メルケプストラム(スペクトル)用 決定木ファイル
tree-pit.inf	基本周波数(F0)用　決定木ファイル
info.txt	話者、コーパスセット情報ファイル

8．2 HTS Training の流れ



*斜体太字であらわされているファイルはHMMファイル類です。

８．３ プログラム処理内容

８．３．１ 拡張環境依存音素ラベルの作成

外部コマンドplinfoにより、言語韻律情報ファイルと音素時間ラベルファイルから拡張環境依存音素ラベルを作成し、音素リストと拡張環境依存音素リストおよび各々のマスターラベルファイルを作成します。

Mklist.java
　（外部コマンド）
　plinfo

input:　…/*.mk　：　音素時間ラベルファイル
　　…/*.pli　：　言語韻律情報ファイル
　　…/data/proto/plinfo-data/ATR_jp_00.kp　仮名音素変換表ファイル

output: …/*.lab
　　…/output/mlf/full.mlf　：　拡張環境依存音素のマスターラベルファイル
　　…/output/mlf/mono.mlf　：　音素のマスターラベルファイル
　　…/output/lst/full.lst　：　拡張環境依存音素リスト
　　…/output/lst/mono.lst　：　音素リスト

マスターラベルファイルについては、HTKのマニュアルを参照してください。

８．３．２ 音響特徴量データファイルの作成

音声ファイルから外部コマンドにより特徴量データファイルを作成します。

Mkdata.java

（外部コマンド）

straight-analysis		:	メルケプストラムの計算
resampf0		:	基本周波数(F0)データのリサンプリング
mk_logf0		:	基本周波数(F0)データの対数変換 log10( )
delta		:	特徴量の変化量(デルタパラメータ)の計算
swab		:	バイトスワップ
merge		:	メルケプストラムと基本周波数データの結合
add_htk_header		:	HTK/HTS 用ヘッダの付加
input:	…/*.wav	:	音声ファイル（音声波形データ）
	…/*.f0	:	基本周波数(F0)データ
output:	…/*.cmp	:	特徴量データファイル

８．３．３ HMM ファイル類の作成

HTS バージョン 1.1.1 のコマンドを利用し、HMM ファイル類を作成します。

（中間ファイル等の記述は省略。Move.java 実行後のファイルを最終出力ファイルとして記述しています。）

Training.java

（外部コマンド）

HCompV

HInit

HRest

HHEd

HERest

input:	…/*.cmp	:	特徴量データファイル
	…/output/mlf/full.mlf		
	…/output/mlf/mono.mlf		
	…/output/lst/full.lst	:	拡張環境依存音素リスト
	…/output/lst/mono.lst	:	音素リスト
output:	…/duration.pdf	:	音素継続時間長　HMM ファイル
	…/mcep.pdf	:	メルケプストラム HMM ファイル
	…/pitch.pdf	:	基本周波数　HMM ファイル
	…/tree-dur.inf	:	音素継続時間長　決定木ファイル
	…/tree-mcep.inf	:	メルケプストラム 決定木ファイル
	…/tree-pit.inf	:	基本周波数　決定木ファイル
	…/mcep_d1.win	:	メルケプストラム 窓ファイル(一次)
	…/mcep_d2.win	:	メルケプストラム 窓ファイル(二次)
	…/pit_d1.win	:	基本周波数　窓ファイル(一次)
	…/pit_d2.win	:	基本周波数　窓ファイル(二次)

9. XIMERA SAPI

9.1 プログラム構成

XIMERA SAPI のソースは、XIMERA SAPI DLL ソースおよび、XIMERA SAPI セットアッププログラムソースの二つのプロジェクトから構成されます。ディレクトリ構成を以下に示します。

```
└─src
    └─ximera-sapi
        ├─SetupXimera   XIMERA SAPI のセットアッププログラムソース
        └─Ximera        XIMERA SAPI DLL ソース
```

９．２ レジストリ

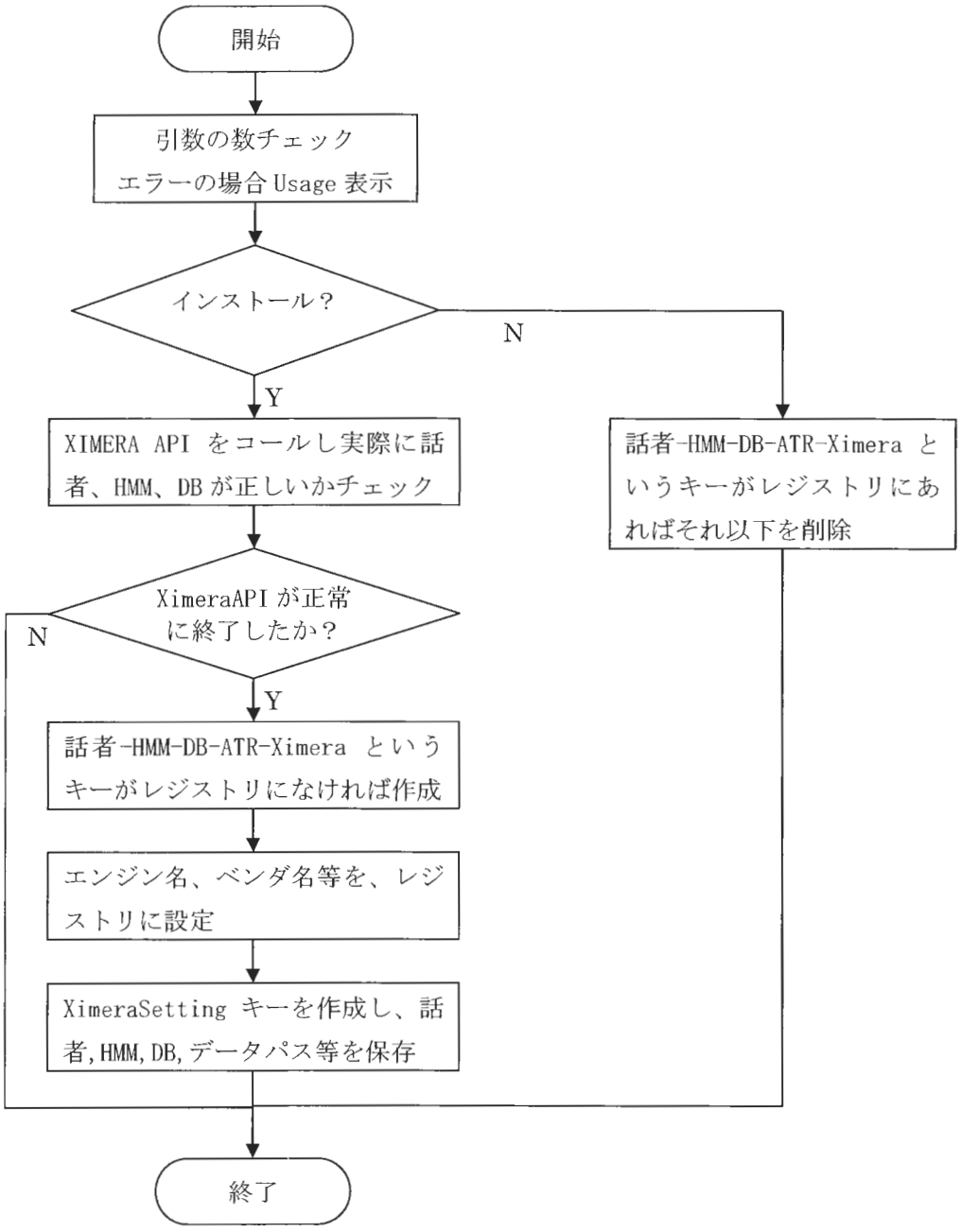
SetupXimera プログラムを使用し XIMERA SAPI のインストールを行うと、SAPI の仕様にしたがい、レジストリに XIMERA の情報が登録されます。ここでは、登録される情報の概要を解説します。

９．２．１ 話者情報

HKEY_LOCAL_MACHINE¥SOFTWARE¥MICROSOFT¥Speech¥Voices¥Tokens 以下に、インストール時に指定した話者･HMM･データベースを元にキーが作成されます。例えば、話者名に F009 、HMM に F009_HTS、データベースに F009_503 をインストール時に指定した場合は、F009-F009_HTS-F009_503-ATR-Ximera というキーが作成されます。このキー以下に SAPI で定義されている情報が格納されます。また、その下には XimeraSetting キーも作成されます。これは、SAPI の仕様に定義されているものではなく、XIMERA SAPI が独自に使用する情報が格納されます。XimeraSetting 以下には音声データのフォーマットの情報、データパス、インストール時に指定した話者名、HMM 名、データベース名が格納されます。

９．３ SetupXimera 処理フロー

SetupXimera は、XIMERA SAPI が必要とする情報並びに SAPI が XIMERA SAPI を認識するための情報をレジストリに登録、または登録した情報を削除します。登録の際には、データパス、話者、HMM、データベースを指定します。これらの情報を引数に XIMERA API の ximera_init メソッド(XIMERA API リファレンス参照)、ximera_speaker メソッドをコールし正常に動作した場合のみ、レジストリに登録します。



9．4 XIMERA SAPI 処理フロー

XIMERA SAPI は、SAPI が定めるメソッドの実処理を行います。XIMERA SAPI では SAPI からコールされたメソッド内で、要求されたイベントに対応した XIMERA API をコールしその結果を SAPI に返します。ここでは、SAPI が定義するメソッド単位で処理概要を説明します。また、SAPI からコールされるメソッドの詳細および、各メソッドが呼び出されるタイミング等は SAPI SDK のマニュアルをご参照ください。

9．4．1 CXimeraTTSEngine

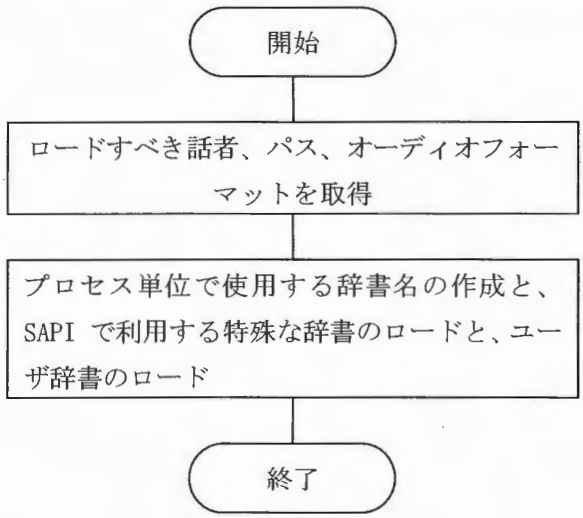
コンストラクタでは、メンバ変数の初期化のみを実行します。

9．4．2 ~CXimeraTTSEngine

SAPI で使用している辞書の削除と、XIMERA の後処理を実行します。

9．4．3 CXimeraTTSEngine::SetObjectToken

SAPI より渡されるレジストリに記録されている情報にもとづき XIMERA を初期化します。また、辞書の初期化も実行します。



9．4．4 CXimeraTTSEngine::GetObjectToken

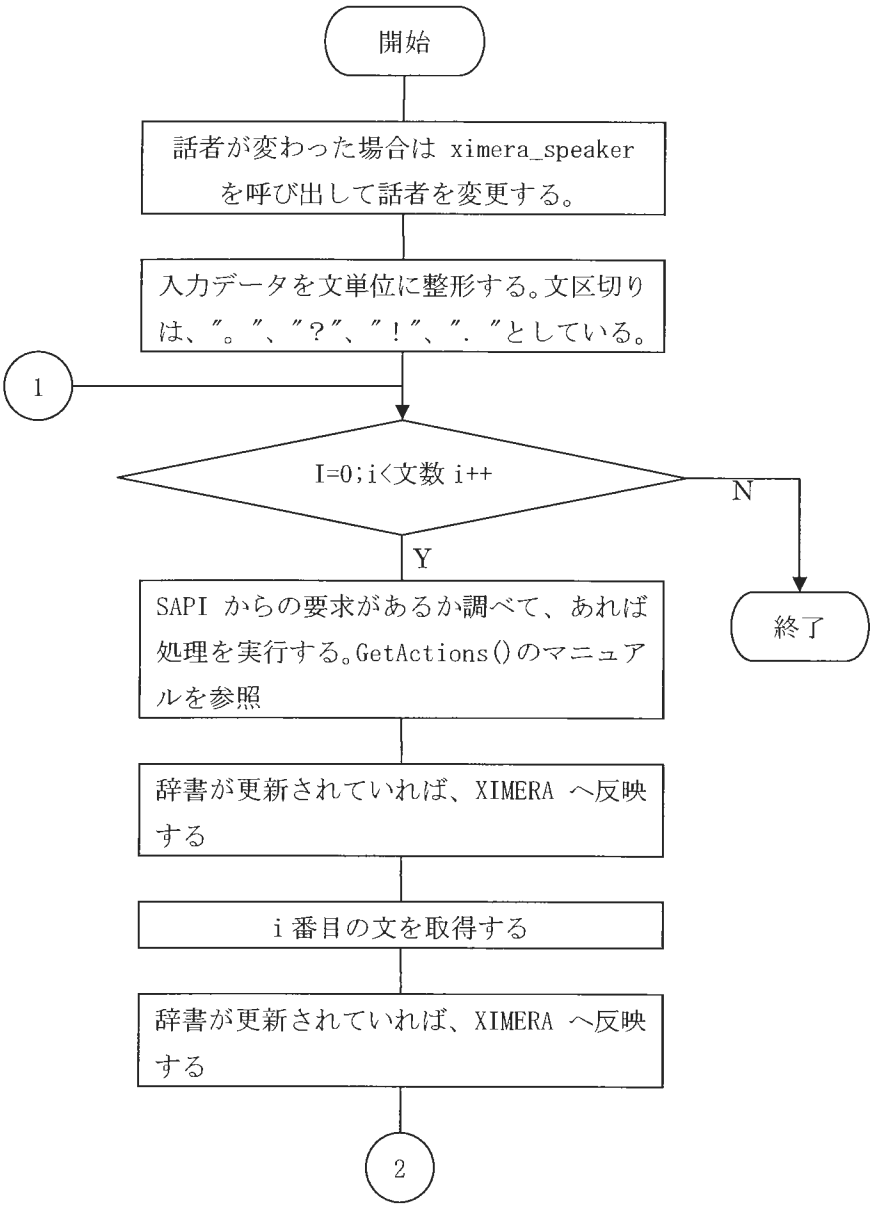
SetObjectToken で保持した IspObjectToken を返します。

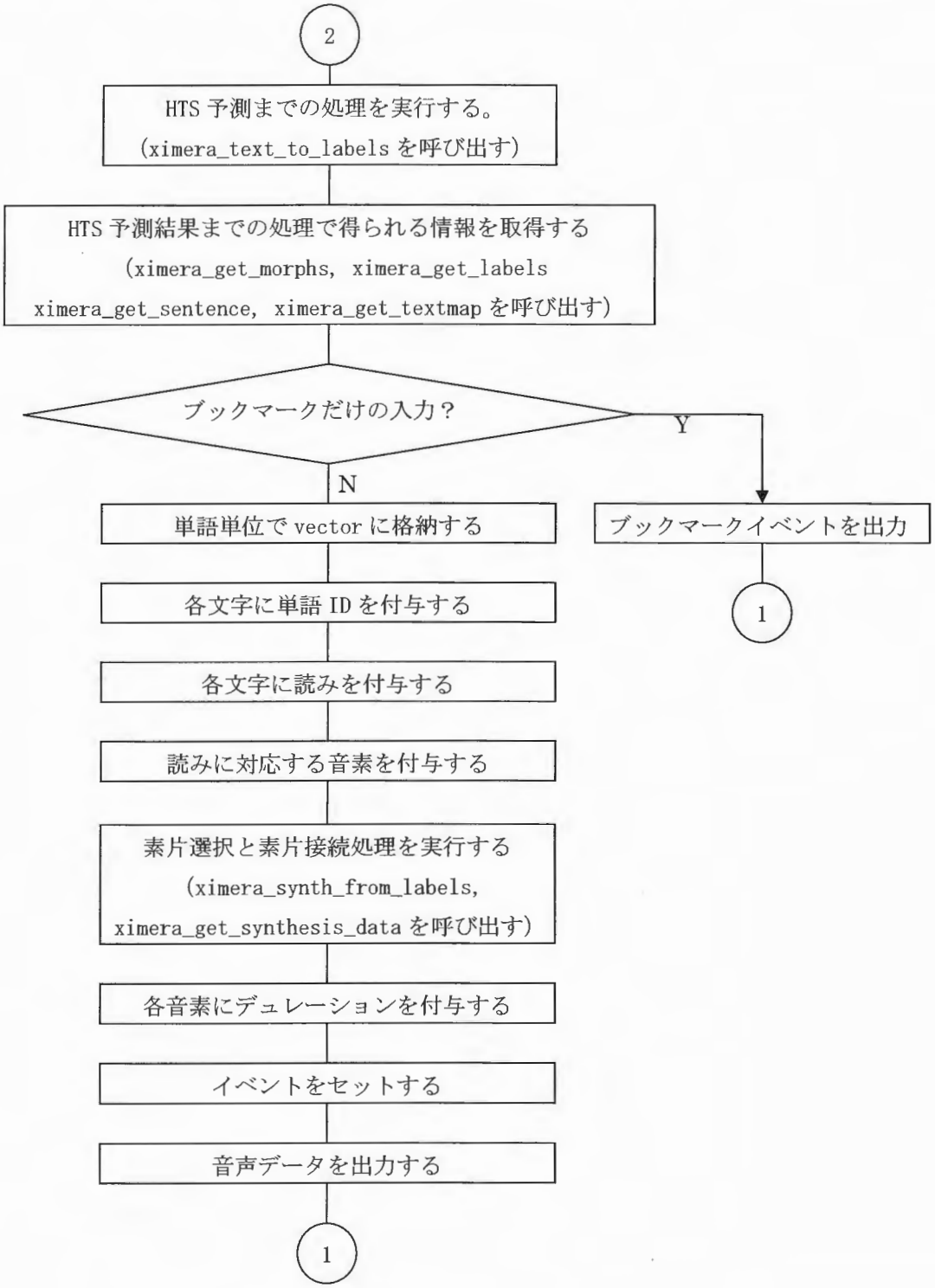
9．4．5 CXimeraTTSEngine::GetOutputFormat

SetObjectToken で保持した出力オーディオフォーマットに基づき、出力フォーマットを返します。

9.4.6 CXimeraTTSEngine::Speak

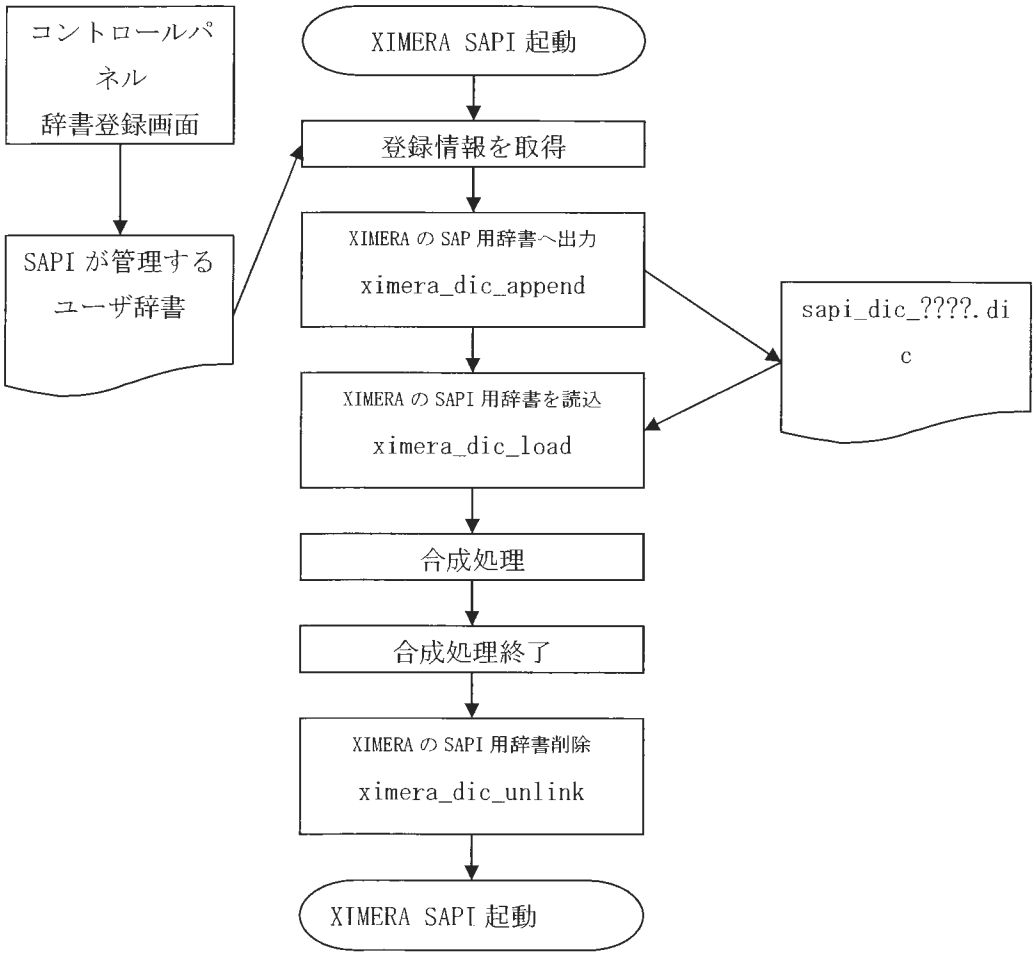
このメソッドが音声合成を行うための主となる関数です。アプリケーションからの合成指示に一対一に対応して呼び出されます。入力テキストは XML タグを解析し、リスト状に変数に格納された状態で引数として渡ってきます。入力にしたがい、音声を合成し、イベントと音声データを出力します。





９．４．７ 辞書登録処理

辞書は、XIMERA のユーザ辞書とは別に、SAPI 用のユーザ辞書が存在します。SAPI 用の辞書登録は、コントロールパネル→音声認識→音声合成の音声の選択で XIMERA が選択されている状態で、設定ボタンを押すことで行えます(XIMERA SAPI ユーザーズマニュアル参照)。この段階では、Windows の SAPI が管理するユーザ辞書に登録されます。登録された辞書は、XIMERA SAPI 起動時、辞書登録されたデータを読込み、データパス¥settings¥windows¥ipadic ディレクトリの下に sapi_dic_????.dic(????はプロセス ID)ファイルを作成し、XIMERA API ximera_dic_append をコールし、登録された辞書内容を出力した後、XIMERA API ximera_dic_load をコールし XIMERA に SAPI 用のユーザ辞書を読み込みます。また、特定のタイミング(9.4.6 CXimeraTTSEngine::Speak の項参照)でコントロールパネル上から、辞書が更新されているかチェックし、更新されていた場合、辞書ファイルを再作成し、XIMERA に辞書を読み込みます。なお、XIMERA SAPI 終了時に SAPI 用の辞書ファイルを削除します。



XIMERA（Ver1.1）

XIMERA API リファレンス

(C/C++用 API 一覧)

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

XIMERA Version1.1 API　一覧

No.	関数名	機　　能
1.1	ximera_init	話者ディレクトリ、settings ディレクトリが置かれているトップディレクトリのパスを指定し、初期化を行います。初期化の際にデータベースに依存しない設定が読み込まれます。
1.2	ximera_speaker	話者を設定します。
1.3	ximera_synth	文字列を入力し、音声合成処理を実行します。
1.4	ximera_cleanup	後処理を実行します。
1.5	ximera_synth_from_plinfo	言語韻律情報を入力し、音声合成処理を実行します。
1.6	ximera_synth_from_labels	拡張環境依存ラベルを入力とし、音声合成処理を実行します。
1.7	ximera_text_to_morphs	仮名漢字文字列から形態素情報への変換を実行します。
1.8	ximera_text_to_plinfo	仮名漢字文字列から言語韻律情報への変換を実行します。
1.9	ximera_text_to_labels	仮名漢字文字列から拡張環境依存ラベルへの変換を実行します。
1.10	ximera_morphs_to_plinfo	形態素情報から言語韻律情報への変換を実行します。
1.11	ximera_plinfo_to_labels	言語韻律情報から拡張環境依存ラベルへの変換を実行します。
1.12	ximera_labels_to_targets	拡張環境依存ラベルから HTS 予測結果を生成します。
1.13	ximera_synth_select	HTS 予測結果から素片選択を実行します。
1.14	ximera_synth_select	素片選択を実行します。
1.15	ximera_synth_connect	素片接続を一括実行します。
1.16	ximera_get_morphs	形態素情報を取得します。
1.17	ximera_get_bancha	数詞句解析結果を付与した形態素情報を取得します。
1.18	ximera_get_bound	境界情報を付与した形態素情報を取得します。
1.19	ximera_get_textmap	テキストのマッピング情報を取得します。
1.20	ximera_get_plinfo	言語韻律情報を取得します。
1.21	ximera_get_labels	拡張環境依存ラベルを取得します。
1.22	ximera_get_sentence	文情報を取得します。
1.23	ximera_get_hts_mk	予測音素長情報を取得します。
1.24	ximera_get_hts_f0	予測ピッチ情報を取得します。
1.25	ximera_get_select	素片選択結果を取得します。
1.26	ximera_get_synthesis_data	音声合成結果を取得します。
1.27	ximera_get_voice_env	音声データフォーマットを取得します。

No.	関数名	機　　能
1.28	ximera_delete_synthesis_data	合成結果の格納された構造体のメモリを開放します。
1.29	ximera_save_morphs	形態素情報をファイルへ出力します。
1.30	ximera_save_bancha	数詞句解析結果付与した形態素情報をファイルへ出力します。
1.31	ximera_save_bound	境界情報を付与した形態素情報をファイルへ出力します。
1.32	ximera_save_cabocha	係り受けを付与した情報をファイルへ出力します。
1.33	ximera_save_textmap	テキストマップ情報をファイルへ出力します。
1.34	ximera_save_plinfo	言語韻律情報をファイルへ出力します。
1.35	ximera_save_sentence	文情報をファイルへ出力します。
1.36	ximera_save_labels	拡張環境依存ラベルをファイルへ出力します。
1.37	ximera_save_hts_mk	予測音素長情報をファイルへ出力します。
1.38	ximera_save_hts_f0	予測ピッチ情報をファイルへ出力します。
1.39	ximera_save_hts_cep	予測メルケプストラムをファイルへ出力します。
1.40	ximera_save_select_seg	素片選択結果をファイルへ出力します。
1.41	ximera_dic_load	ユーザ辞書をロードします。
1.42	ximera_dic_unload	ユーザ辞書をアンロードします。
1.43	ximera_dic_unlink	ユーザ辞書を削除します。
1.44	ximera_dic_append	ユーザ辞書に単語を追加します。
1.45	ximera_dic_batch_append	ユーザ辞書に単語をバッチで追加します。
1.46	ximera_set_duration_stretch	話速の調整を行います。
1.47	ximera_set_pitch_stretch	ピッチの調整を行います。
1.48	ximera_set_prun_cost_width	コスト計算時のパラメータを調整します。
1.49	ximera_set_prun_num_candidate	コスト計算時のパラメータを調整します。
1.50	ximera_set_prun_num_cost	コスト計算時のパラメータを調整します。

XIMERA（Ver1.1）

ユーザーズマニュアル

= GUI =

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

本マニュアルは、ATR であらたに開発されたコーパスベース方式による音声波形素片接続型テキスト音声合成エンジン（開発コード名：XIMERA、以下「XIMERA」と記す）を実行する GUI（Graphical User Interface）ベースのサンプルアプリケーションの操作手順を説明します。以下このサンプルアプリケーションを「 XIMERA GUI 」と記述します。

XIMERA GUIはJavaで書かれていますので、あらかじめJavaが実行できる環境（J2SE™ v1.4.2_04）がインストールされている必要があります。

操作方法は、Windows 版・Linux 版とも基本的に同じです。
異なる場合は、■Linux 版■として表記していますのでご注意ください。

※ 操作方法を実際にお試しいただく際に必要なサンプルデータと出力ファイル例を以下にご用意しています。
Windows 版と Linux 版があります。文字コード（SJIS・EUC）が異なります。
ご利用ください。

Ximera110¥sample¥ximera-gui

また、最終章で音声合成の高速化に利用できる「頻度情報作成ツール」の使用方法を説明します。

１．機能概要

1) XIMERA GUI メインウィンドウ上に入力したテキスト、またはテキストファイル（.txt）をロードし合成音声を作成し再生します。

2) XIMERA GUI メインウィンドウ上に入力したテキストの、選択範囲部分のみ合成音声を作成し再生します。

3) 再生した合成音声を wav ファイルとして保存します。

4) あらかじめ用意されたテキストファイル（.txt）からバッチ処理で合成音声を作成し、wav ファイルとして保存します。

5) 設定を変更することにより、話速・ピッチを任意に変更できます。

6) 設定を変更することにより、予備選択時のコスト幅・素片候補数を任意に変更できます。

7) 設定を変更することにより、テンポラリーディレクトリを任意のディレクトリに変更できます。

8) 設定を変更することにより、中間データをファイルとして出力・保存します。
（14 種類）

9) 言語韻律情報ファイル（.pli）・拡張環境依存ラベルファイル（.lab）をロードし合成音声を作成し再生します。

10) HTS（HMM-based Speech Synthesis System）予測結果から合成音声を作成し再生します。

11) あらかじめ用意された言語韻律情報ファイル（.pli）・拡張環境依存ラベルファイル（.lab）からバッチ処理で合成音声を作成し、wav ファイルとして保存します。

12) 表示を設定することにより、中間データファイルの内容をサブウィンドウで表示することができます。（15 種類）

13) 既存の辞書以外に、ユーザー辞書の登録・削除ができ、辞書を指定してロードできます。

2．操作説明

2．1 起動方法

XIMERA GUI は JNI (Java Native Interface) を用いて XIMERA ライブラリを操作しますので、XimeraForJNI.dll がパスの通った位置に置かれているか、カレントディレクトリに置かれている必要があります。コマンドプロンプトから GUI を起動する方法は以下のとおりです。

java -Xmx512m -Xss10m -jar ..¥Ximera.jar

（ Ximera110¥bin¥windows に移動し、xiui.bat から起動できます。）

図 2．1 コマンドプロンプト画面

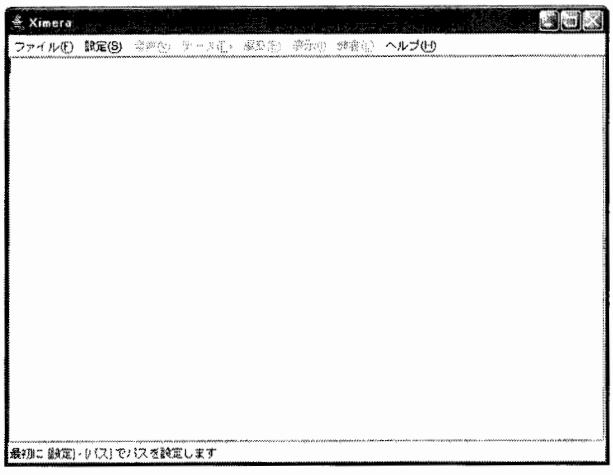


図 2．2 XIMERA GUI メインウィンドウ

■Linux 版■

java -Xmx512m -Xss10m -jar ../Ximera.jar
（ Ximera110/bin/linux　に移動し、xiui.sh から起動できます。）

※起動前に、環境変数 LD_LIBRARY_PATH に Ximera110/bin/linux が追加されているか環境変数設定をご確認ください。
＜詳細はインストールマニュアルを参照してください。＞

2．2 設定

2．2．1 データベース（話者）・HTS 学習結果の設定

① GUI を起動後まず、データディレクトリへのパスを設定する必要があります。
XIMERA GUI メインウィンドウメニューバーの【設定（S)】[パス（Ctrl-G）] メニューを実行し、表示されたデータディレクトリ選択画面で、xdata/ ディレクトリを指定してください。

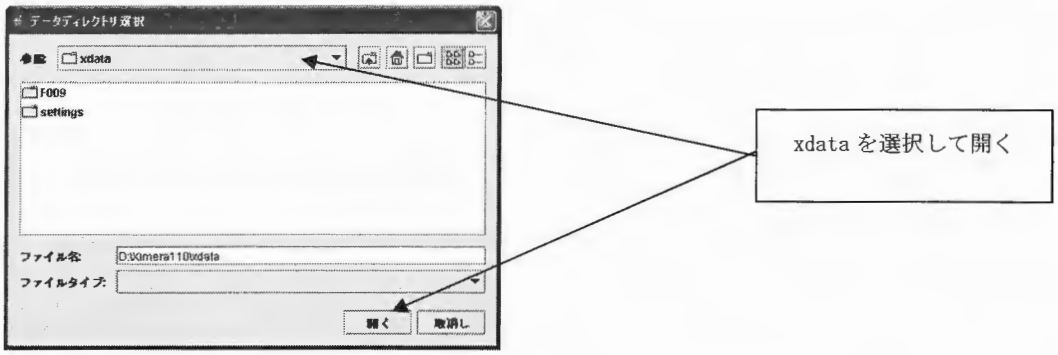


図 2．3 データディレクトリ選択画面

② データディレクトリを指定して開き、表示されたデータベース選択画面でデータベース（話者）を選択します。
デフォルトで「F009」が選択されますので、必要に応じて切り替えてください。

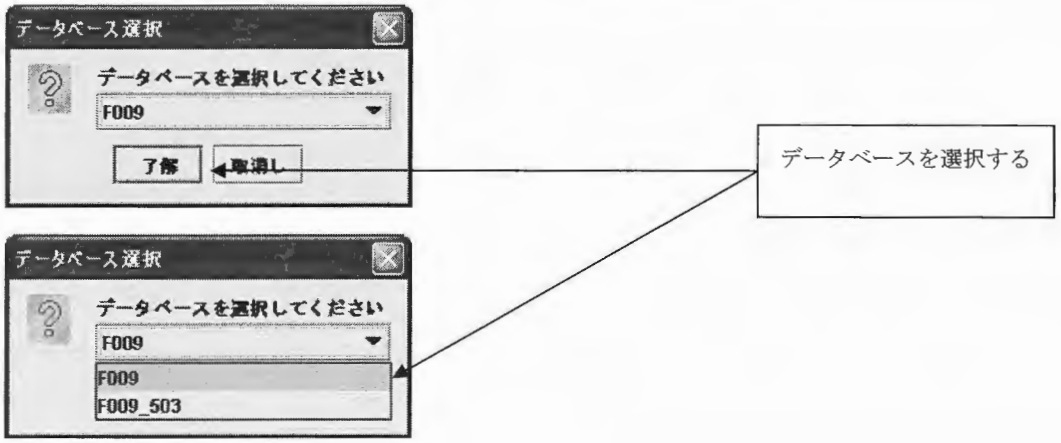


図 2．4 データベース選択画面

※XIMERA Version1.1 の話者は女性のみで、大きさの異なる 2 つのデータベースが用意されています。
F009（正味 47.6 時間の音声ファイル）と F009_503（正味 34.5 分の音声ファイル）です。

③　韻律パラメータ生成用の HMM を指定します。

図 2.5　HTS 選択画面

※XIMERA Version1.1 で配布する話者モデルは、“F009_hts” の 1 つだけなので、上記選択画面が出ることなく自動的にロードされ、次の工程へ進みます。

④　ステータスバーのメッセージが、
「DB をロードしています」→「DB をロードしています...完了」に変わるとメニューバーのすべてのコマンドが有効になります。

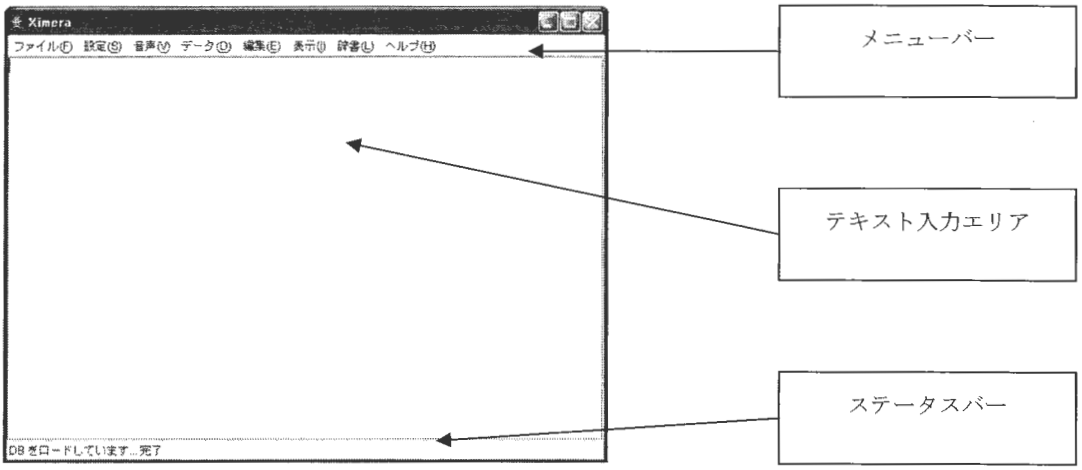


図 2.6　データベース・ HTS 設定後の XIMERA GUI メインウィンドウ

＜注意＞
　データベースのロードに失敗する場合は、茶筌・南瓜の設定が正しく行われていない可能性があります。
詳細は、＜インストールマニュアル 茶筌・南瓜設定＞を参照してください。

⑤ 最初に選択したデータベースを変更したい場合は、XIMERA GUI メインウィンドウメニューバーの【設定（S)】［データベースの切り替え]メニューを実行し、データベースを切り替えてください。

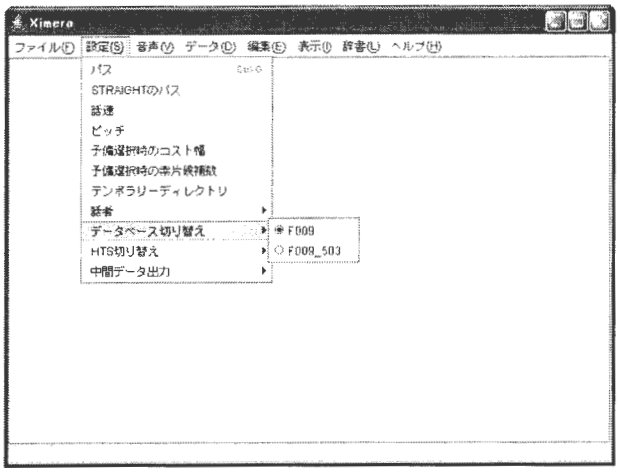


図 2.7 データベース設定変更画面

2．2．2 STRAIGHTの設定

HTS予測結果から合成音声を生成する場合、予測結果の確認用として
「 straight-synthesis.exe 」を外部コマンドとして利用します。
ここで、「 straight-synthesis.exe 」へのパスを設定します。

XIMERA GUI メインウィンドウメニューバーの【設定 (S)】[STRAIGHT のパス] メニューを実行し、表示された出力ファイル名の入力画面から「 straight-synthesis.exe 」を選択し、パスを設定します。
XIMERA Version1.1の場合は、Ximera110¥bin¥windows に置かれています。

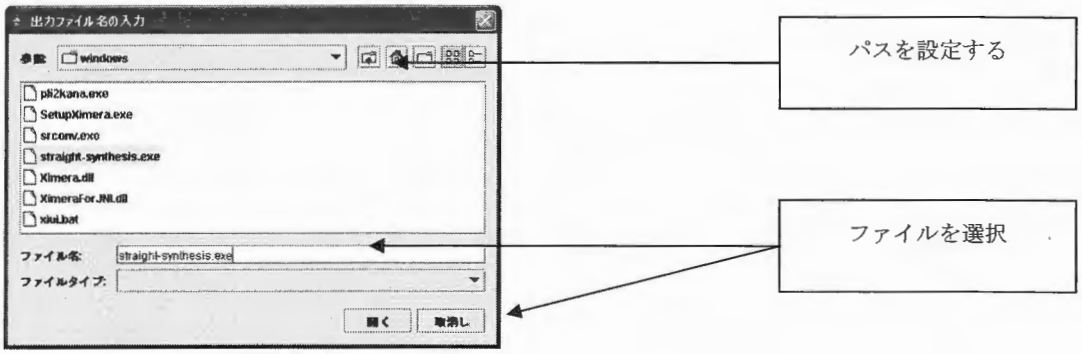


図 2．8 STRAIGHTのパス設定画面

■Linux版■

「 straight-synthesis 」を外部コマンドとして利用します。
ここで、「 straight-synthesis 」へのパスを設定します。
XIMERA GUI メインウィンドウメニューバーの【設定 (S)】[STRAIGHT のパス] メニューを実行し、表示された出力ファイル名の入力画面から「 straight-synthesis 」を選択し、パスを設定します。
XIMERA Version1.1の場合は、Ximera110/bin/linux に置かれています。

２．２．３ 話速・ピッチの設定

話速・ピッチの設定数値（倍率）を変更することにより、出力される合成音声を調整することができます。

ただし、HTS 予測結果を設定数値（倍率）で修正し調整するため、実際に出力される合成音声が指定どおりに調整されない場合があります。

① 話速

XIMERA GUI メインウィンドウメニューバーの【設定（S)】[話速] メニューを実行し、表示された話速の設定画面に任意の数値を入力します。

(入力可能な数値は 0.5 から 5.0 となっています。)

デフォルト値の 1.0 を 大きくすると話速が遅くなり、小さくすると話速が速くなります。

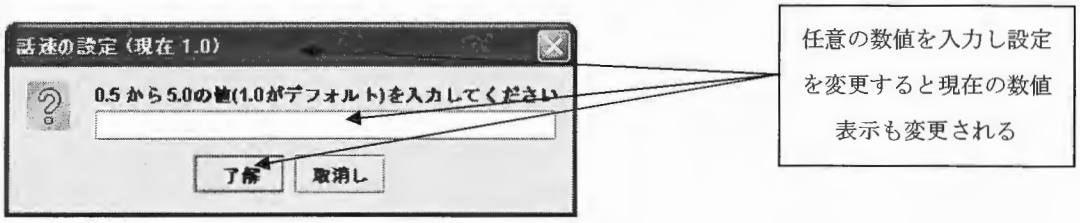


図 ２．９ 話速の設定画面

② ピッチ

XIMERA GUI メインウィンドウメニューバーの【設定（S)】[ピッチ] メニューを実行し、表示されたピッチの設定画面に任意の数値を入力します。

(入力可能な数値は 0.5 から 5.0 となっています。)

デフォルト値の 1.0 を 大きくするとピッチが高くなり、小さくするとピッチが低くなります。

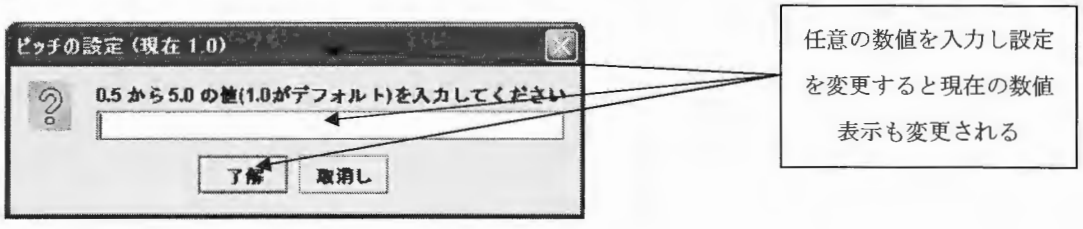


図 ２．１０ ピッチの設定画面

２．２．４ 予備選択時のコスト幅・素片候補数の設定

コスト幅・素片候補数の設定を変更することにより、素片選択時の計算時間と音質を調整できます。

① コスト幅

XIMERA GUI メインウィンドウメニューバーの【設定（S)】[予備選択時のコスト幅] メニューを実行し、表示されたコスト幅の設定画面に任意の数値を入力します。
予備選択を行う際にコスト幅で、「 最低コスト値 ＋ 設定値 」 の範囲外の素片を、素片予備選択候補から外します。
（デフォルト値は 5.0）

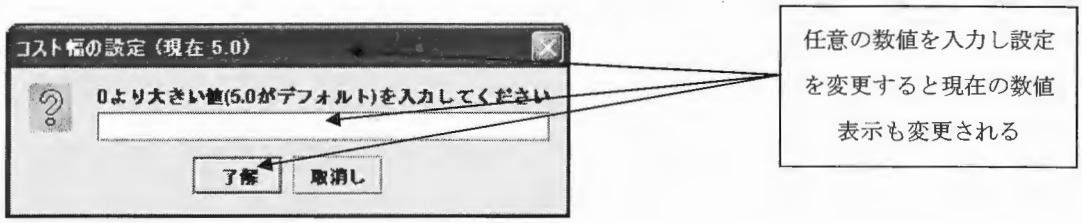


図 ２．１１ コスト幅の設定画面

② 予備選択時の素片候補数

XIMERA GUI メインウィンドウメニューバーの【設定（S)】[予備選択時の素片候補数] メニューを実行し、表示された予備選択時の素片候補数設定画面に任意の数値を入力します。
ここでの設定値以上の候補がある場合は、予備選択が実行されます。
（デフォルト値は 1600）

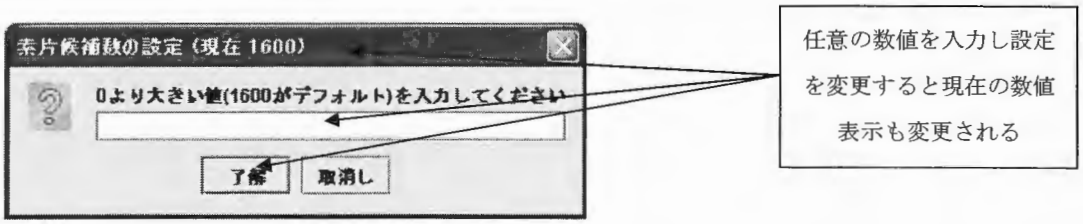


図 ２．１２ 予備選択時の素片候補数設定画面

※できるだけ基本設定（デフォルト値）でのご使用をお奨めします。
予備選択の詳細については＜プログラマーズマニュアル 予備選択＞を参照してください。

２．２．５ テンポラリーディレクトリの設定

XIMERA GUI で作成された合成音声ファイル（.wav）や各種中間データファイルを設定により出力・保存した場合（中間データ出力設定については後述）、カレントディレクトリに一時的に保存されます。
この各種テンポラリーファイルを、設定により任意のディレクトリに保存できます。

XIMERA GUI メインウィンドウメニューバーの【設定(S)】[テンポラリーディレクトリ] メニューを実行し、表示されたテンポラリーディレクトリの選択画面で、任意のディレクトリのパスを選択するか、ディレクトリを新規作成します。

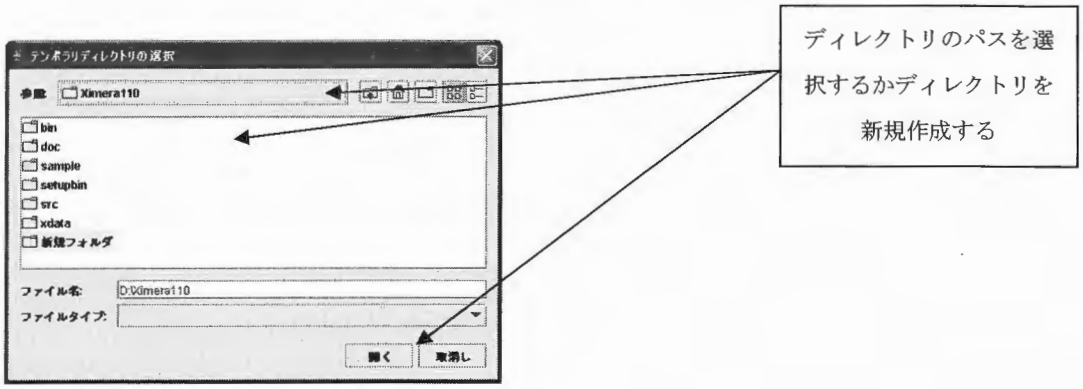


図 ２．１３ テンポラリーディレクトリ選択画面

２．２．６ 中間データ出力の設定

合成音声を生成するために必要な各種中間データを、ファイルとして出力・保存するかどうかを設定できます。

XIMERA GUI メインウィンドウメニューバーの【設定（S)】[中間データ出力] メニューを実行すると中間データメニューが表示されます。

ファイルとして出力・保存したい中間データのチェックボックスに「チェックマーク」を入れます。

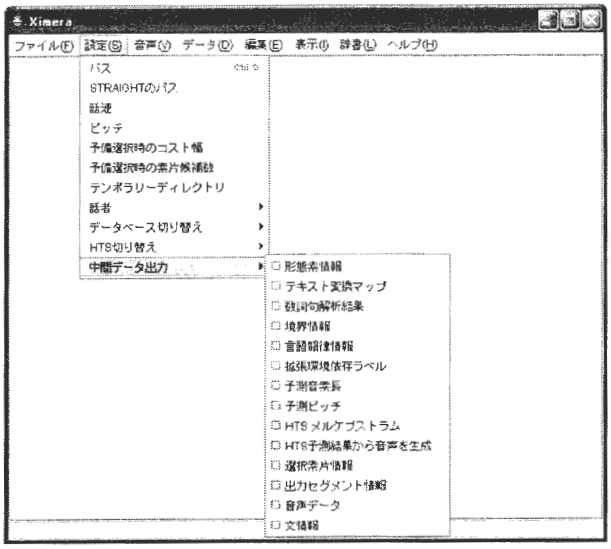


図 ２．１４ 中間データファイル出力・保存設定画面

中間データの出力・保存ファイル名は、音声合成方法（テンポラリー合成 または バッチ合成）によって異なります。バッチ合成時の出力ファイル名は、<2.3.2 音声の合成（バッチ処理合成）>を参照してください。

表 ２．１ 中間データの出力ファイル名（テンポラリー合成）

	中間データメニュー	出力・保存ファイル名
①	形態素情報	tmp_synth.wav.mor
②	テキスト変換マップ	tmp_synth.wav.map
③	数詞句解析結果	tmp_synth.wav.num
④	境界情報	tmp_synth.wav.bou
⑤	言語韻律情報	tmp_synth.wav.pli
⑥	拡張環境依存ラベル	tmp_synth.wav.lab
⑦	予測音素長	tmp_synth.wav.mk
⑧	予測ピッチ	tmp_synth.wav.f0
⑨	HTS メルケプストラム	tmp_synth.wav.cep
⑩	HTS 予測結果から音声を生成	tmp_synth.wav.hts.wav

⑪	選択素片情報	tmp_synth.wav.seg
⑫	出力セグメント情報	tmp_synth.wav.outseg
⑬	音声データ	tmp_synth.wav
⑭	文情報	tmp_synth.wav.sen

表 2.2 中間データメニューの概要

	中間データメニュー	概要
①	形態素情報（.mor）	茶筌から出力される形態素情報
②	テキスト変換マップ（.map）	入力テキストを形態素解析用テキストに変換
③	数詞句解析結果（.num）	形態素情報に数詞句解析結果を反映させた情報
④	境界情報（.bou）	形態素情報に自動ポーズ挿入のための境界情報を反映させた情報
⑤	言語韻律情報（.pli）	①から生成される言語韻律情報
⑥	拡張環境依存ラベル（.lab）	①から生成される拡張環境依存ラベル情報
⑦	予測音素長（.mk）	予測結果の音素長
⑧	予測ピッチ（.f0）	予測結果のF0情報
⑨	HTSメルケプストラム（.cep）	予測結果のメルケプストラム情報
⑩	HTS予測結果から音声を生成（.hts.wav）	⑧⑨から生成された合成音声
⑪	選択素片情報（.seg）	選択された素片情報
⑫	出力セグメント情報（.outseg）	素片接続された音素と音素長
⑬	音声データ（.wav）	生成された合成音声
⑭	文情報（.sen）	入力テキストと予測結果の音素との対応情報

ファイル内容の具体的な表示例については、<2.6 表示> を参照してください。

２．３ 音声

２．３．１ 音声の合成（テンポラリー合成）

① 音声の合成と再生

XIMERA GUI メインウィンドウのテキスト入力エリアに、音声合成したいテキストを入力し、メニューバーの【音声 (V)】[音声の合成と再生 (Ctrl-P)] メニューを実行します。合成音声が生成され再生されます。

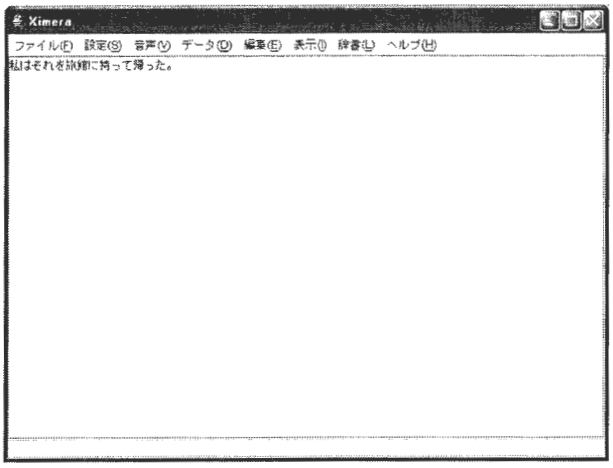


図 ２．１５ 音声の合成テキスト入力例画面

② もう一度聞く

再度同じ合成音声を再生する場合は、XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[もう一度聞く (Ctrl+Alt-P)] メニューを実行します。

③ 選択範囲の合成と再生

テキスト入力エリアに入力したテキストの一部分のみを音声合成したい場合は、合成する箇所をマウス左ボタンでドラッグして選択し、XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[選択範囲の合成と再生 (Ctrl+Shift-P)] メニューを実行します。

④ 言語韻律情報から音声を合成して再生

言語韻律情報から合成音声を生成し再生する場合は、XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[言語韻律情報から音声を合成して再生] メニューを実行し、言語韻律情報ファイルの選択画面から音声合成したいファイル (.pli) を選択します。

＜注意＞

ロードするファイルは、文字コードが Windows 版：SJIS、Linux 版：EUC で記述されていることをご確認ください。

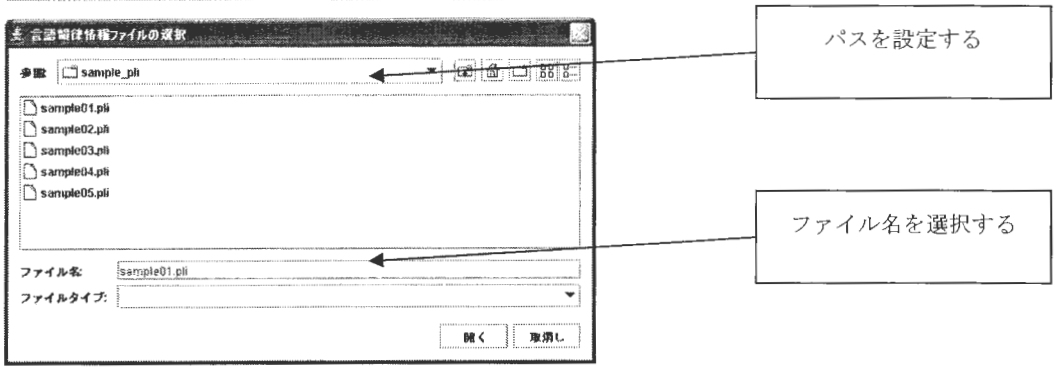


図 2．16 言語韻律情報ファイルの選択画面

⑤ ラベルから音声を合成して再生

拡張環境依存ラベルから合成音声を生成し再生する場合は、XIMERA GUI メインウィンドウメニューバーの【音声（V)】[ラベルから音声を合成して再生] メニューを実行し、ラベルファイルの選択画面から音声合成したいファイル（.lab）を選択します。

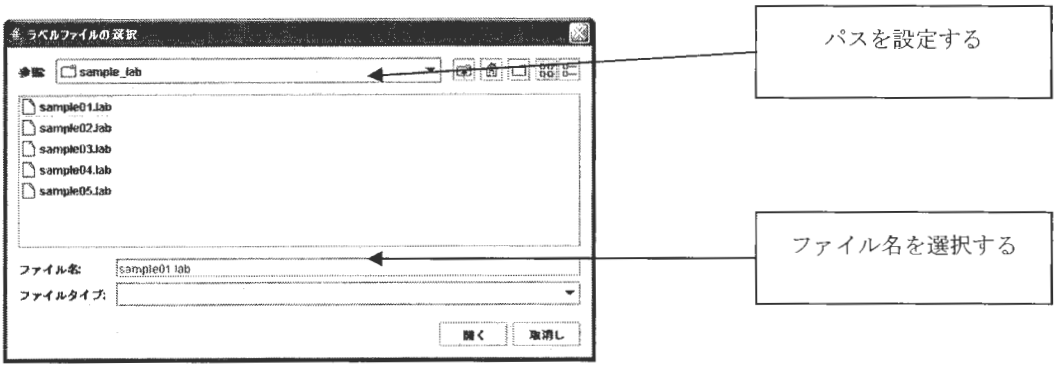


図 2．17 ラベルファイルの選択画面

※1：ここで再生される合成音声は、メモリ上にある合成音声です。

※2：テンポラリー合成の合成音声・中間データは、設定によりファイルとして出力・保存することができます。

設定方法は、<2.2.6 中間データ出力の設定>を参照してください。

※3：<2.6 表示>で出力音声波形または出力音声スペクトログラムを表示するように設定してあると、表示画面から合成音声の全体または一部を再生することができます。

<2.6.11 出力音声波形、2.6.13 出力音声スペクトログラム>を参照してください。

２．３．２ 音声の合成（バッチ処理合成）

① テキストファイル（.txt）からのバッチ処理合成

テキストファイル（.txt）をロードし合成音声を生成しファイル（.wav）に出力・保存します。

XIMERA GUI メインウィンドウメニューバーの【音声（V)】[バッチ合成（Ctrl-B）] メニューを実行し、バッチファイルの選択画面からバッチ処理で音声合成したいファイル（.txt）を選択します。

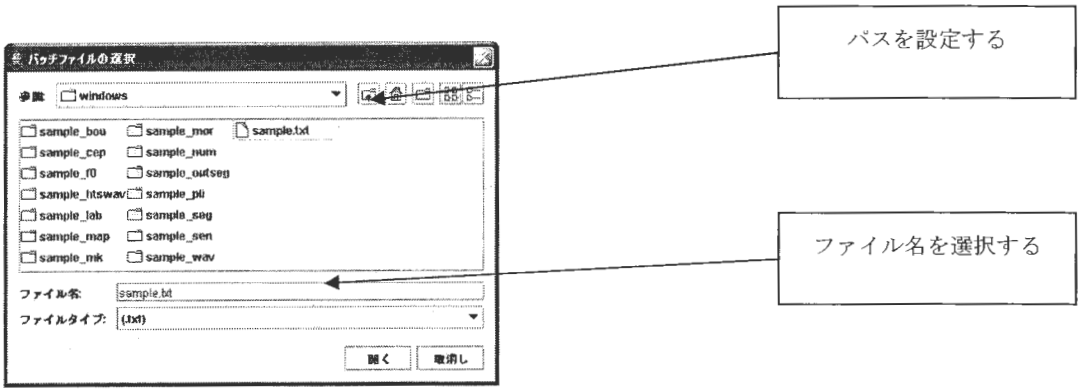


図 ２．１８ バッチファイル（.txt）選択画面

出力先のディレクトリ選択画面から合成音声ファイル（.wav）を出力・保存するディレクトリを指定します。

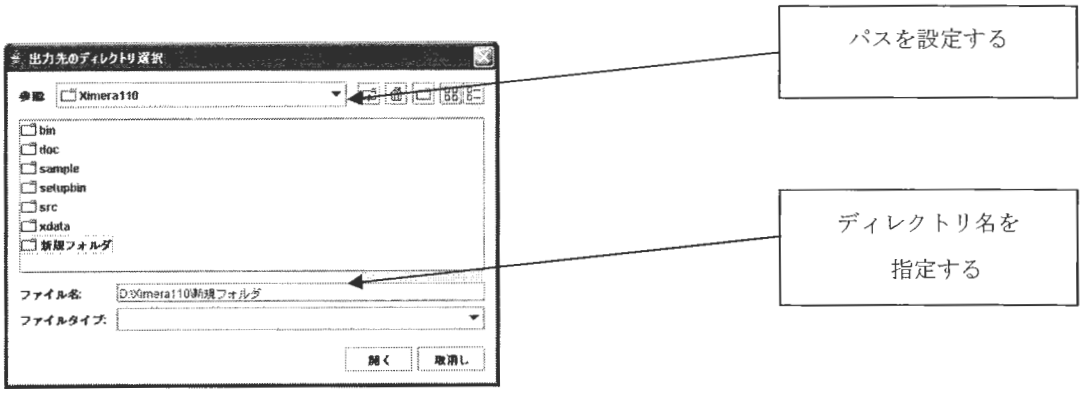


図 ２．１９ 出力先ディレクトリ選択画面

バッチ処理合成が開始され、ステータスバーに現在処理中のテキスト内容が表示されます。

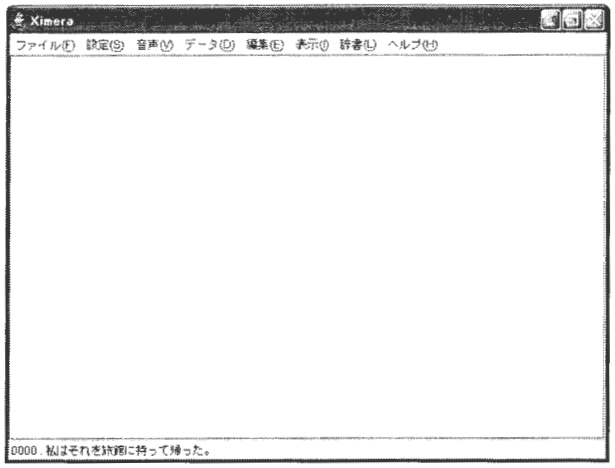


図 2.20 テキストファイル（.txt）からのバッチ処理合成画面

終了するとステータスバーに音声合成にかかった処理時間の log メッセージが表示されます。

図 2.21 テキストファイル（.txt）からのバッチ処理合成終了画面

※1：出力・保存される合成音声ファイル（.wav）は、ロードしたテキストファイル内容の一行が1ファイルになります。

※2：ステータスバーに表示される処理時間の log メッセージは、バッチ処理合成終了後ファイル（.log）として保存されます。1 ファイルごとの音声合成処理時間と全処理時間が記載されています。

※3：中間データをファイルとして出力・保存する設定にしている場合、同じディレクトリに出力・保存されます。

◎出力ファイル名について

出力ファイル名は、「0000、0001、0002…」というように連番で自動的に付与されます。

表 2.3 中間データの出力ファイル名（バッチ処理合成）

	中間データメニュー	出力・保存ファイル名
①	形態素情報	0000.wav.mor
②	テキスト変換マップ	0000.wav.map
③	数詞句解析結果	0000.wav.num
④	境界情報	0000.wav.bou
⑤	言語韻律情報	0000.wav.pli
⑥	拡張環境依存ラベル	0000.wav.lab
⑦	予測音素長	0000.wav.mk
⑧	予測ピッチ	0000.wav.f0
⑨	HTS メルケプストラム	0000.wav.cep
⑩	HTS 予測結果から音声を生成	0000.wav.hts.wav
⑪	選択素片情報	0000.wav.seg
⑫	出力セグメント情報	0000.wav.outseg
⑬	音声データ	0000.wav
⑭	文情報	0000.wav.sen

ファイル内容の具体的な表示例については、<2.6 表示> を参照してください。

処理時間 log のファイル名は、ロードしたテキストファイル名と同じになります。
図 2.18 の表示例の場合は、「sample.txt.log」となっています。

```
---------- 0000 ----------
私はそれを旅館に持って帰った。
処理時間 = 1855 ms
音声長　 = 2407 ms
リアルタイムの 0.771 倍
---------- 0001 ----------
畑は干上がり、土は割れる。
処理時間 = 1699 ms
音声長　 = 2022 ms
リアルタイムの 0.84 倍
---------- 0002 ----------
フロリダ州のゴルフ場で、ホールインワンを達成する。
処理時間 = 2401 ms
音声長　 = 3399 ms
リアルタイムの 0.706 倍
---------- 0003 ----------
健康づくり計画のひとつに、成人の喫煙率を減らすことがあげられる。
処理時間 = 3773 ms
音声長　 = 4694 ms
リアルタイムの 0.804 倍
---------- 0004 ----------
日米間の貿易不均衡拡大を背景に、要求は９分野５４項目と昨年を上回り、対日強硬姿勢を一段と鮮明とした。
処理時間 = 7016 ms
音声長　 = 10280 ms
リアルタイムの 0.682 倍

---------- 全処理に対する結果 ----------
処理時間 = 16744 ms
音声長　 = 22802 ms
リアルタイムの 0.734 倍
```

図 2.22 sample.txt.log ファイルの内容

② 言語韻律情報ファイル（.pli）からのバッチ処理合成

言語韻律情報ファイル（.pli）をロードし合成音声を生成しファイル（.wav）に出力・保存します。

テキストファイル（.txt）からのバッチ処理合成と異なり、ロードする言語韻律情報ファイル（.pli）1ファイルから合成音声ファイル（.wav）1ファイルを生成し保存します。

XIMERA GUI メインウィンドウメニューバーの【音声(V)】[言語韻律情報バッチ合成] メニューを実行します。

言語韻律情報が置かれているディレクトリ選択画面から音声合成したいファイル（.pli）のあるディレクトリを選択します。

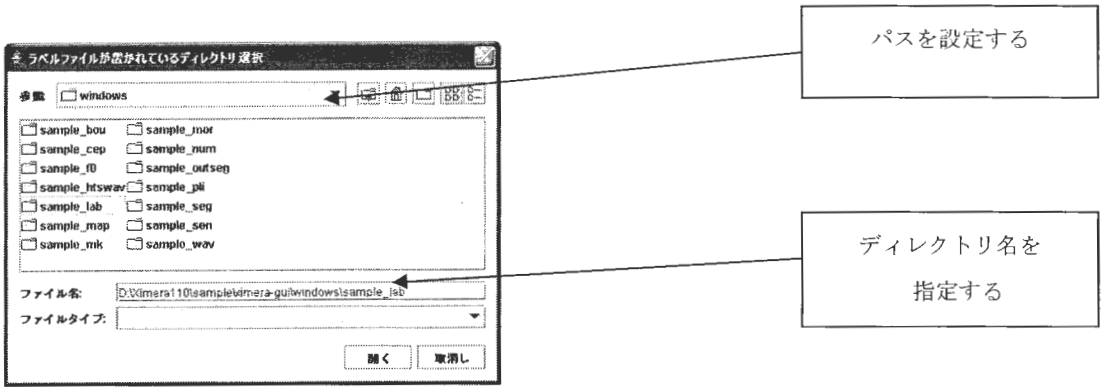


図 2.23 言語韻律情報ファイル（.pli）選択画面

出力先のディレクトリ選択画面から合成音声ファイル（.wav）を出力・保存するディレクトリを指定します。

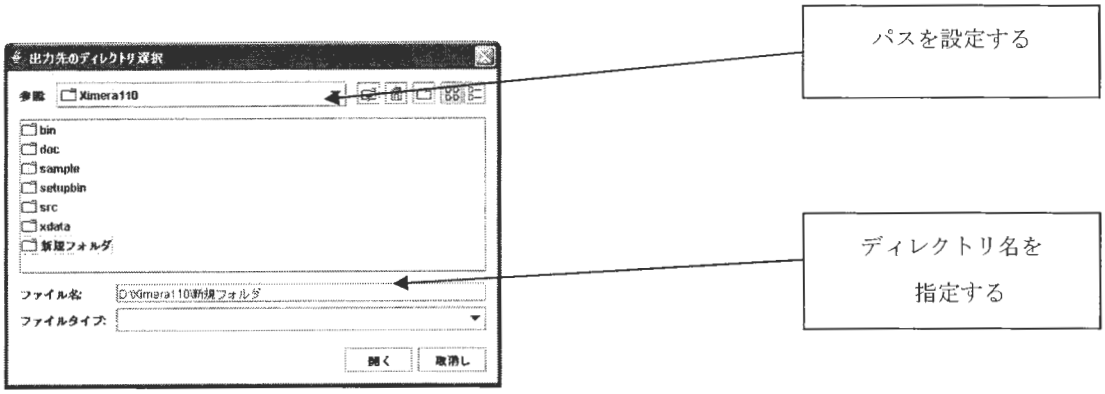


図 2.24 出力先ディレクトリ選択画面

バッチ処理合成が開始され、ステータスバーに現在処理中のファイル名が表示されます。

図 2.25 言語韻律情報ファイル（.pli）からのバッチ処理合成画面

図 2.26 言語韻律情報ファイル（.pli）からのバッチ処理合成終了画面

※1：テキストファイル（.txt）からのバッチ処理合成と異なり、音声合成にかかった処理時間の log メッセージは表示されません。

※2：中間データをファイルとして出力・保存する設定にしている場合、同じディレクトリに出力・保存されます。

※3：出力ファイル名は、ロードしたファイル名に関わらず連番で自動的に付与されます。出力ファイル名は ＜表 2.3 中間データの出力ファイル名（バッチ処理合成）＞ を参照してください。

＜注意＞

ロードするファイルは、文字コードが Windows 版：SJIS、Linux 版：EUC で記述されていることをご確認ください。

③ 拡張環境依存ラベルファイル（.lab）からのバッチ処理合成

拡張環境依存ラベルファイル（.lab）をロードし合成音声を生成しファイル（.wav）に出力・保存します。

テキストファイル（.txt）からのバッチ処理合成と異なり、ロードする拡張環境依存ラベルファイル（.lab）1ファイルから合成音声ファイル（.wav）1ファイルを生成し保存します。

XIMERA GUI メインウィンドウメニューバーの【音声（V)】[ラベルバッチ合成] メニューを実行します。

拡張環境依存ラベルファイルが置かれているディレクトリ選択画面から音声合成したいファイル（.lab）のあるディレクトリを選択します。

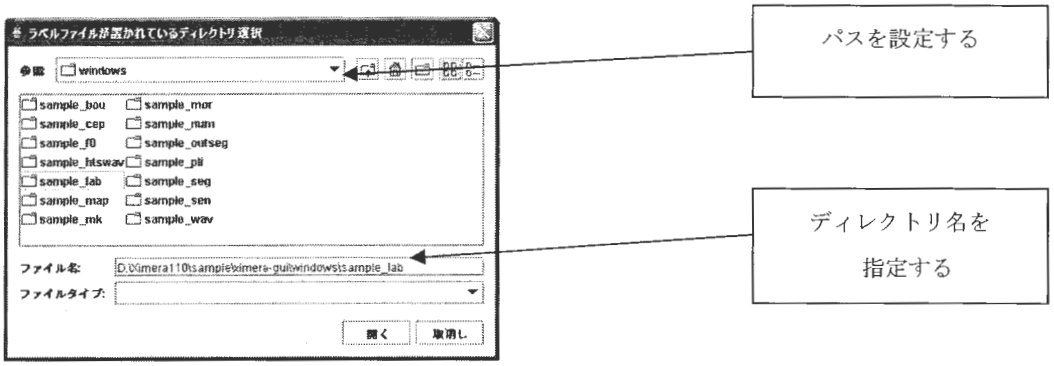


図 2.27 拡張環境依存ラベルファイル（.lab）選択画面

出力先のディレクトリ選択画面から合成音声ファイル（.wav）を出力・保存するディレクトリを指定します。

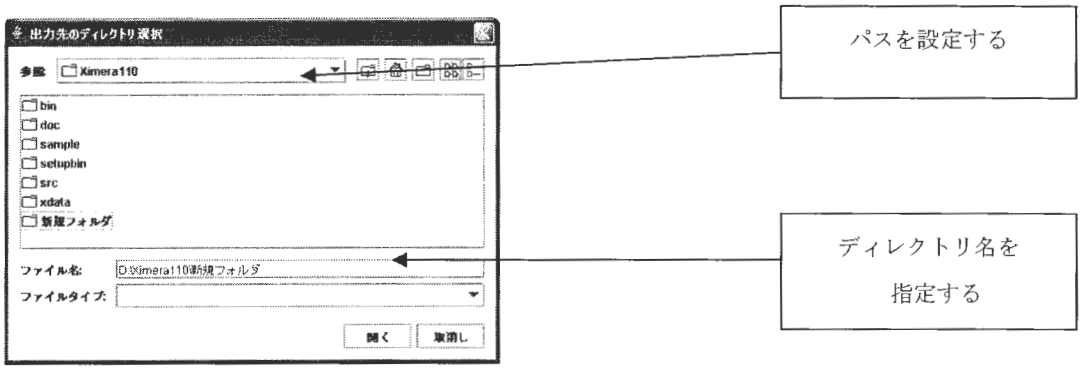


図 2.28 出力先ディレクトリ選択画面

バッチ処理合成が開始され、ステータスバーに現在処理中のファイル名が表示されます。

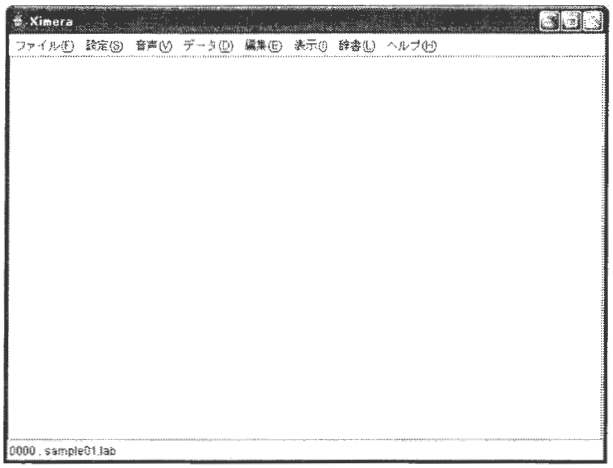


図 2．2９ 拡張環境依存ラベル（.lab）からのバッチ処理合成画面

図 2．３０ 拡張環境依存ラベル（.lab）からのバッチ処理合成終了画面

※1：テキストファイル（.txt）からのバッチ処理合成と異なり、音声合成にかかった処理時間の log メッセージは表示されません。

※2：中間データをファイルとして出力・保存する設定にしている場合、同じディレクトリに出力・保存されます。

※3：出力ファイル名は、ロードしたファイル名に関わらず連番で自動的に付与されます。出力ファイル名は ＜表 2.3 中間データの出力ファイル名（バッチ処理合成）＞ を参照してください。

２．３．３ HTS 予測結果の再生

HTS 予測結果のデータを使用し、外部コマンドとして「 straight-synthesis.exe 」を利用して合成音声を生成し再生できます。

予測結果の確認用として、また XIMERA GUI で生成した合成音声との比較用として利用できます。

ただし、以下の手順が必要です。

① STRAIGHT のパスを設定する。
（<2.2.2 STRAIGHT の設定>を参照してください。）

② 音声合成に必要な中間データファイルを出力するように設定する。
（<2.2.6 中間データ出力の設定>を参照してください。）

※「HTS 予測結果から音声を生成」のチェックボックスにチェックマークを入れると、音声合成に必要な中間データのメニューが自動的に選択されますので便利です。

③ XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[音声の合成と再生 (Ctrl-P)] メニューを実行する。

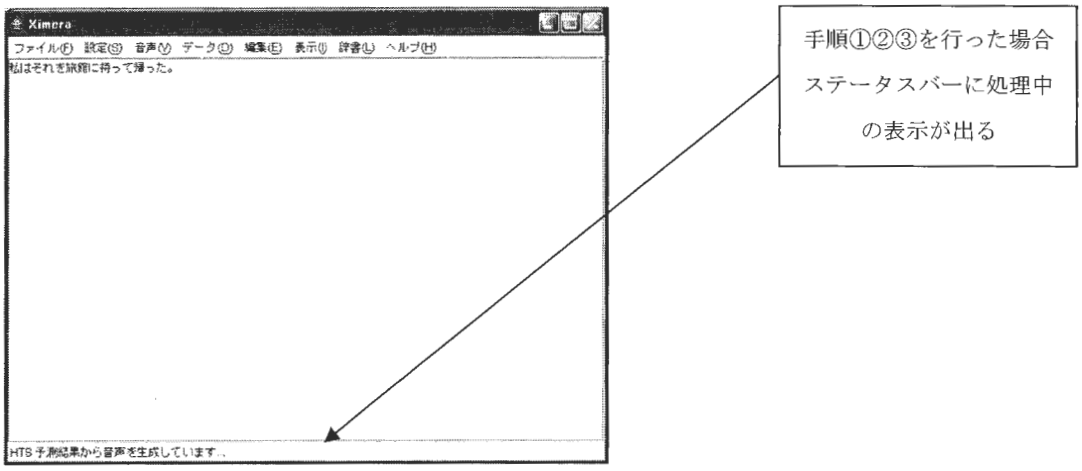


図 ２．３１ HTS 予測結果からの音声合成画面

ステータスバーのメッセージが、

「HTS 予測結果から音声を生成しています」→

「HTS 予測結果から音声を生成しています...完了」に変わってから XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[HTS 予測結果の再生] メニューを実行します。合成音声が再生されます。

２．３．４ 音声の保存

① 合成音声に任意のファイル名をつけて、保存することができます。
テキストを入力して XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[ファイルに保存 (Ctrl-Q)] メニューを実行します。
表示された出力ファイル名入力画面で、出力先ディレクトリのパスを選択し、任意のファイル名を入力します。

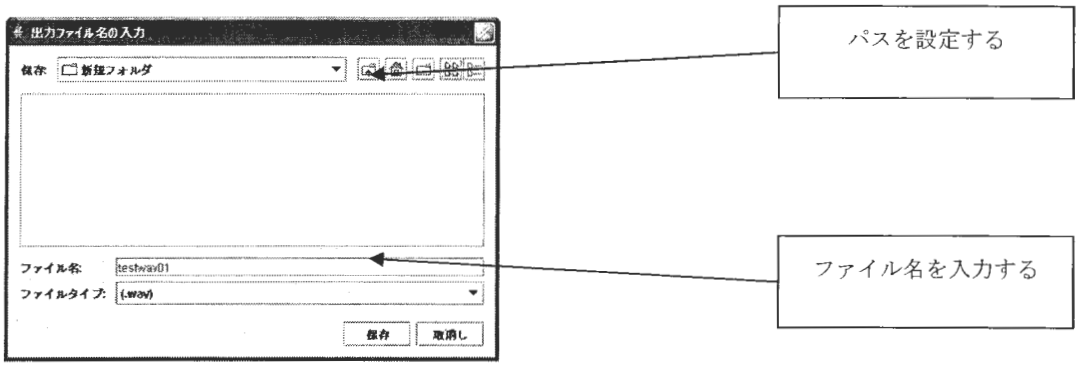


図 ２．３２ 合成音声の出力先・ファイル名指定画面

保存を押下すると、音声の合成と再生が行われて指定ディレクトリに保存されます。

② 選択範囲の合成音声に任意のファイル名をつけて保存する時も同様です。
テキストを入力して合成する箇所をマウス左ボタンでドラッグして選択し、XIMERA GUI メインウィンドウメニューバーの【音声 (V)】[選択範囲をファイルに保存 (Ctrl+Shift-Q)] メニューを実行します。
表示された出力ファイル名入力画面で、出力先ディレクトリのパスを選択し、任意のファイル名を入力します。

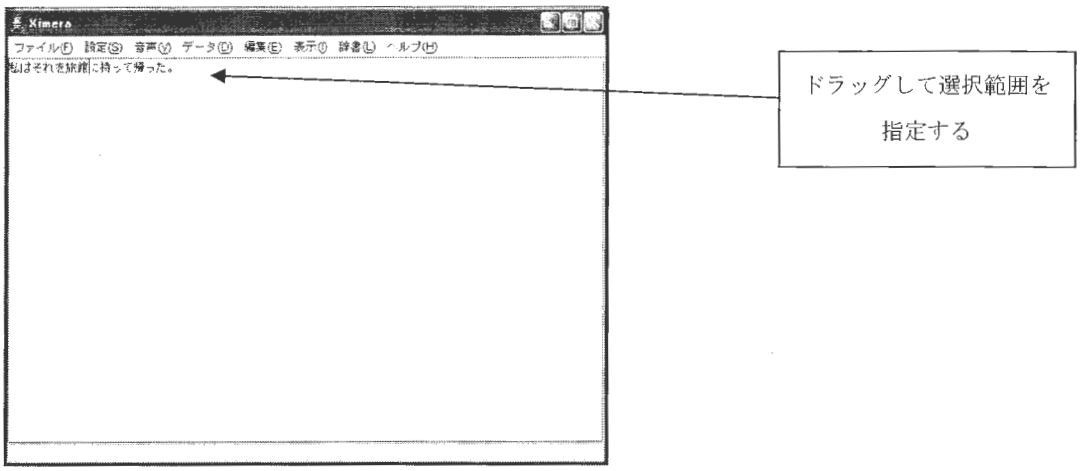


図 2.33 テキスト選択範囲指定画面

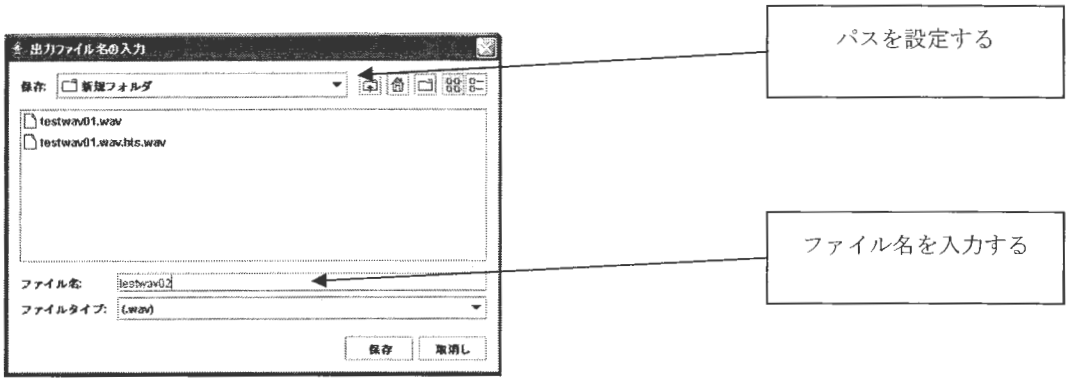


図 2.34 合成音声の出力先・ファイル名指定画面

※1：中間データをファイルとして出力・保存する設定にしている場合、同じディレクトリに出力・保存されます。

※2：中間データのファイル名は任意に指定したファイル名と同じになります。

2．4 テキスト

2．4．1 テキストファイルのロード

① 既存のテキストファイル（.txt）をロードして、テキスト入力エリアに表示します。
XIMERA GUI メインウィンドウメニューバーの【ファイル（F)】[開く（Ctrl-O）] メニューを実行します。
表示されたテキストファイルの選択画面でテキストファイルを選択します。

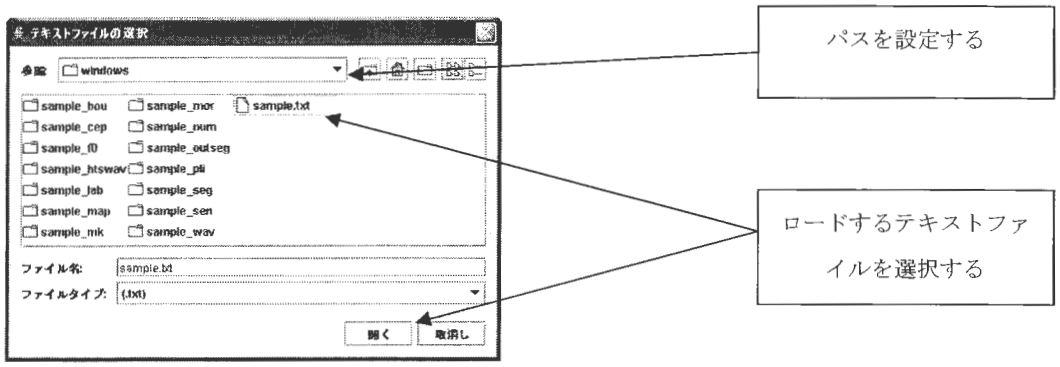


図 2．3 5 テキストファイルの選択画面

② ロードしたテキストファイルが入力エリアに表示されます。

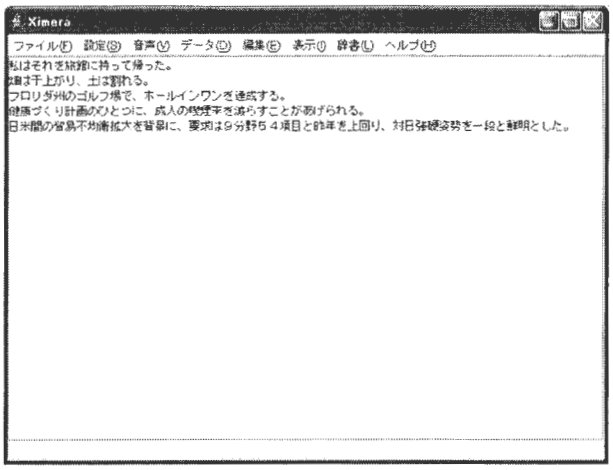


図 2．3 6 テキストファイルロード後画面

２．４．２ テキストの保存

① XIMERA GUI メインウィンドウのテキスト入力エリアに入力したテキストを、テキストファイル（.txt）として保存できます。
メニューバーの【ファイル（F)】[上書き保存（Ctrl-S）] メニューを実行します。
[上書き保存（Ctrl-S）] では、ロードしたテキストファイルまたは、直前に保存したテキストファイルに上書きでテキストを保存します。

② メニューバーの【ファイル（F)】[名前を付けて保存（Ctrl+Shift-S）] メニューを実行した場合は、出力ファイル名の入力画面で、保存するディレクトリを指定して、出力ファイル名を入力し保存します。

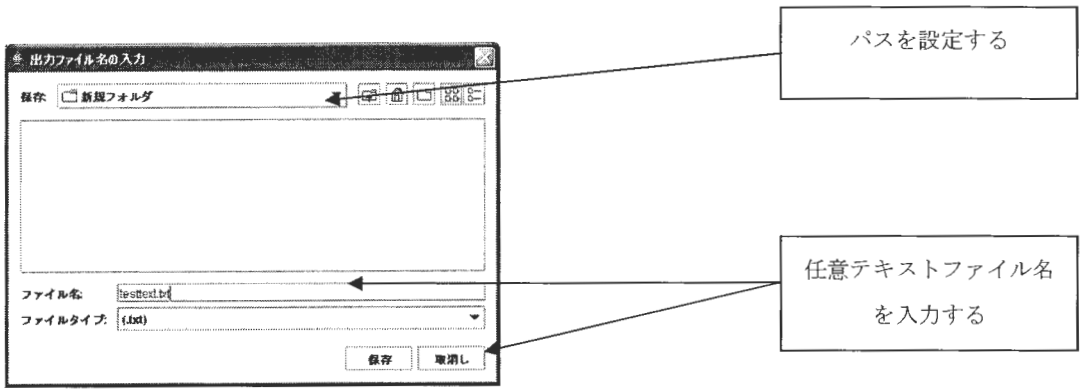


図 ２．３７ 出力ファイル名の入力画面

２．４．３ テキストのクリア

XIMERA GUI メインウィンドウのテキスト入力エリアに表示されているテキストを、すべて消去します。
メニューバーの【編集（E)】[クリア（F2）] メニューを実行します。

２．５ 中間ファイル間の変換

各種中間ファイルから、異なった段階の中間ファイルを作成することができます。ただし、XIMERA で扱えるファイルフォーマットであることが前提となります。

２．５．１ テキストファイルから形態素情報ファイルを作成・保存

XIMERA GUI メインウィンドウメニューバーの【データ (D)】[バッチファイルから形態素情報] メニューを実行します。

バッチファイルの選択画面で、ロードするテキストファイル（.txt）を選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

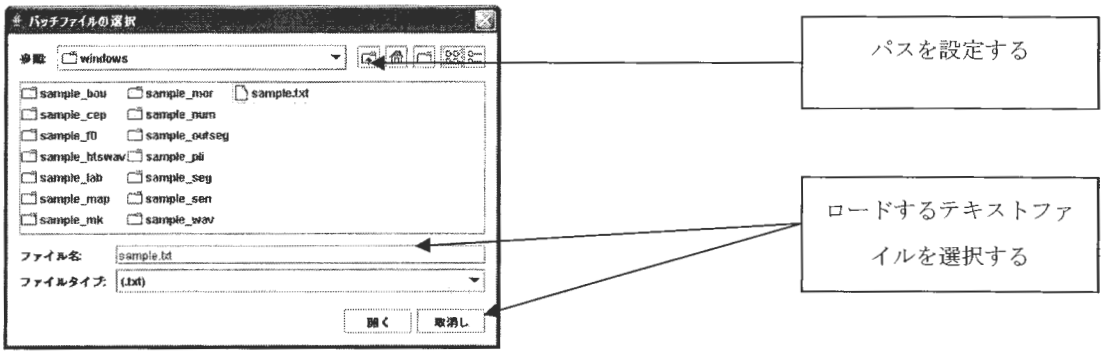


図 ２．３８ テキストファイル（.txt）選択画面

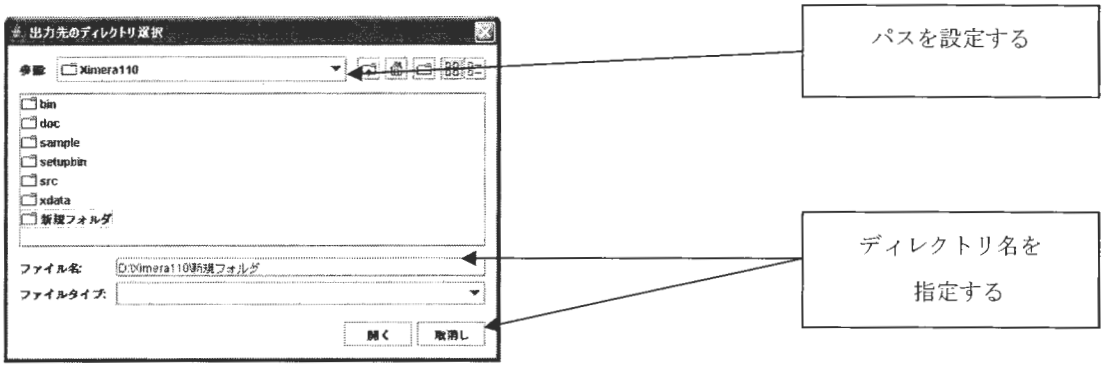


図 ２．３９ 出力先ディレクトリ選択画面

バッチ処理が開始され、ステータスバーに現在処理中のテキスト内容が表示されます。終了するとステータスバーに「形態素情報の出力完了」メッセージが表示されます。

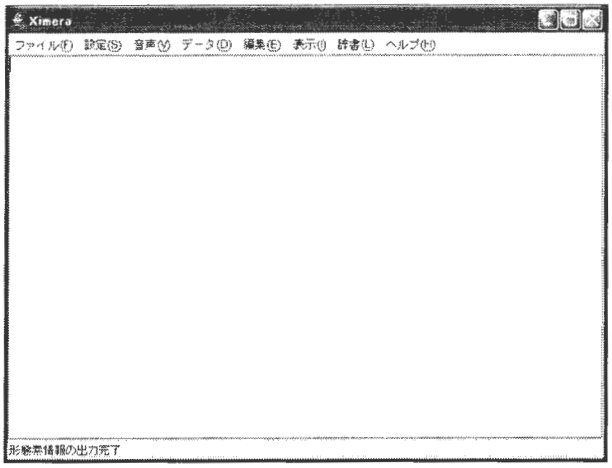


図 2.40 形態素情報出力・保存終了画面

※1：出力・保存される形態素情報ファイル（.mor）は、ロードしたテキストファイル内容の一行が1ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.mor、0001.mor、0002.mor …」というように自動的に付与されます。

２．５．２ テキストファイルから言語韻律情報ファイルを作成・保存

XIMERA GUI メインウィンドウメニューバーの【データ (D)】[バッチファイルから言語韻律情報] メニューを実行します。

バッチファイルの選択画面で、ロードするテキストファイル（.txt）を選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

バッチ処理が開始され、ステータスバーに現在処理中のテキスト内容が表示されます。終了するとステータスバーに「言語韻律情報の出力完了」メッセージが表示されます。

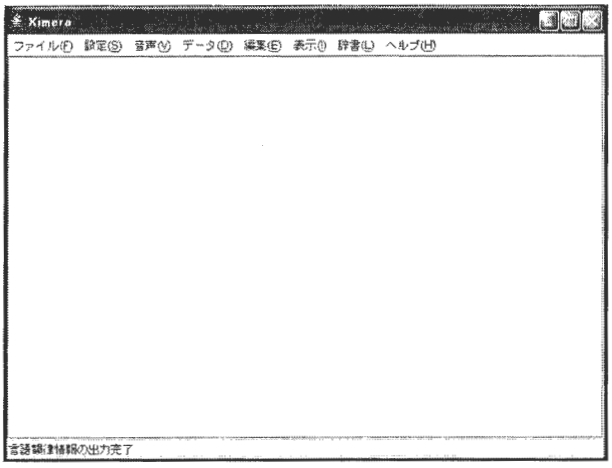


図 ２．４１ 言語韻律情報出力・保存終了画面

※1：出力・保存される言語韻律情報ファイル（.pli）は、ロードしたテキストファイル内容の一行が１ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.pli、0001.pli、0002.pli …」というように自動的に付与されます。

２．５．３ テキストファイルから拡張環境依存ラベルファイルを作成・保存

XIMERA GUI メインウィンドウメニューバーの【データ（D)】[バッチファイルから拡張環境依存ラベル] メニューを実行します。

バッチファイルの選択画面で、ロードするテキストファイル（.txt）を選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

バッチ処理が開始され、ステータスバーに現在処理中のテキスト内容が表示されます。終了するとステータスバーに「拡張環境依存ラベルの出力完了」メッセージが表示されます。

図 ２．４２ 拡張環境依存ラベル出力・保存終了画面

※1：出力・保存される拡張環境依存ラベルファイル（.lab）は、ロードしたテキストファイル内容の一行が１ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.lab、0001.lab、0002.lab …」というように自動的に付与されます。

２．５．４ 形態素情報ファイルから言語韻律情報ファイルを作成・保存

XIMERA GUI メインウィンドウメニューバーの【データ (D)】[形態素情報から言語韻律情報] メニューを実行します。

形態素情報が置かれているディレクトリの選択画面で、ロードする形態素情報ファイル（.mor）が置かれているディレクトリを選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

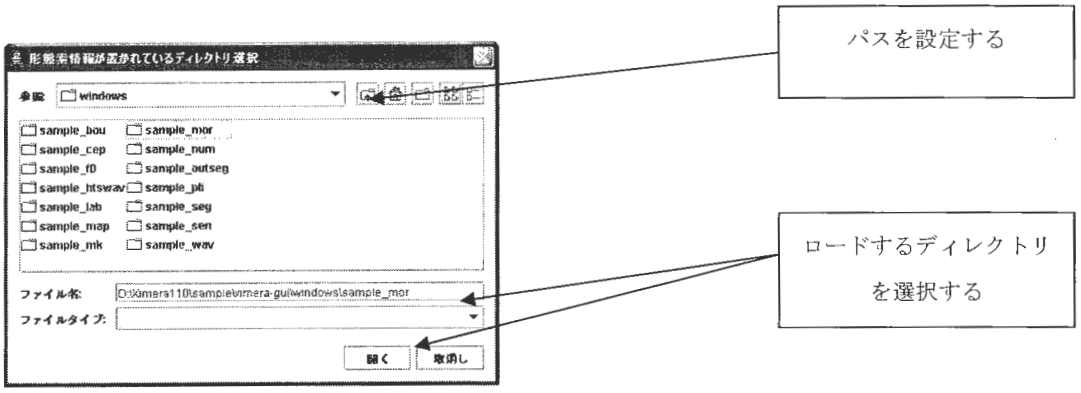


図 ２．４３ 形態素情報ファイルの置かれているディレクトリ選択画面

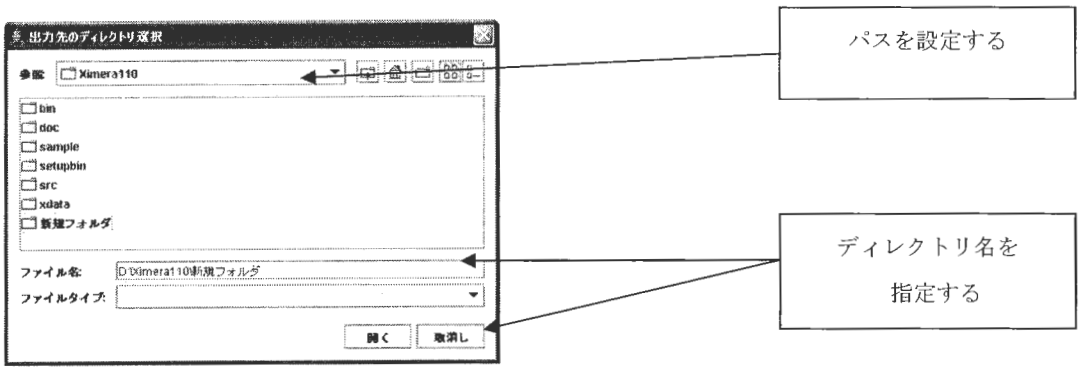


図 ２．４４ 出力先ディレクトリ選択画面

バッチ処理が開始され、ステータスバーに現在処理中のファイル名が表示されます。
終了するとステータスバーに「言語韻律情報の出力完了」メッセージが表示されます。

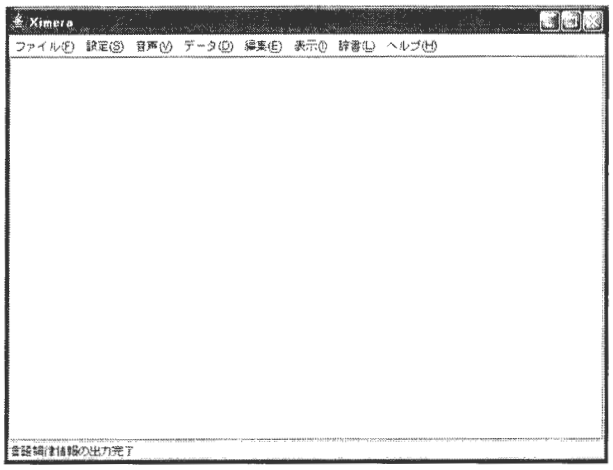


図 2.45 言語韻律情報ファイル出力・保存完了画面

※1：出力・保存される言語韻律情報ファイル（.pli）は、ロードした形態素情報ファイル（.mor）1ファイルに対して1ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.pli、0001.pli、0002.pli …」というように自動的に付与されます。

＜注意＞

ロードするファイルは、文字コードが Windows 版：SJIS、Linux 版：EUC で記述されていることをご確認ください。

２．５．５ 言語韻律情報ファイルから拡張環境依存ラベルファイルを作成・保存

XIMERA GUI メインウィンドウメニューバーの【データ (D)】[言語韻律情報から拡張環境依存ラベル] メニューを実行します。

言語韻律情報が置かれているディレクトリの選択画面で、ロードする言語韻律情報ファイル（.pli）が置かれているディレクトリを選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

バッチ処理が開始され、ステータスバーに現在処理中のファイル名が表示されます。終了するとステータスバーに「拡張環境依存ラベルの出力完了」メッセージが表示されます。

図 ２．４６ 拡張環境依存ラベル出力・保存完了画面

※1：出力・保存される拡張環境依存ラベルファイル（.lab）は、ロードした言語韻律情報ファイル（.pli）１ファイルに対して１ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.lab、0001.lab、0002.lab …」というように自動的に付与されます。

＜注意＞

ロードするファイルは、文字コードが Windows 版：SJIS、Linux 版：EUC で記述されていることをご確認ください。

２．５．６ 拡張環境依存ラベルファイルから HTS 予測結果情報を作成・保存

＜ HTS 予測結果情報とは…＞

予測音素長ファイル(.mk)・予測ケプストラムファイル(.cep)・予測ピッチファイル(.f0)の３種のファイルをさします。

XIMERA GUI メインウィンドウメニューバーの【データ (D)】[ラベルからターゲット情報]メニューを実行します。

拡張環境依存ラベルが置かれているディレクトリの選択画面で、ロードする拡張環境依存ラベルファイル（.lab）が置かれているディレクトリを選択し、出力先のディレクトリ選択画面で、保存するディレクトリを指定します。

バッチ処理が開始され、ステータスバーに現在処理中のファイル名が表示されます。
終了するとステータスバーに「ターゲット情報の出力完了」メッセージが表示されます。

図 ２．４７　HTS 予測結果の出力・保存完了画面

※1：出力・保存される予測音素長ファイル（.mk）・予測ケプストラムファイル（.cep）・予測ピッチファイル（.f0）は、ロードした拡張環境依存ラベルファイル（.lab）1 ファイルに対して各 1 ファイルになります。

※2：出力ファイル名は、ロードしたファイル名に関わらず連番で「0000.mk、0000.cep、0000.f0、0001.mk、0001.cep、0001.f0 …」というように自動的に付与されます。

２．６ 表示

XIMERA GUI で音声の合成（テンポラリー合成）を行った場合、設定により中間データファイルの内容をサブウィンドウで表示することができます。

XIMERA GUI メインウィンドウメニューバーの【表示（I)】メニューを実行すると、サブウィンドウで表示することのできる中間データメニューが表示されます。
サブウィンドウで表示したい中間データのチェックボックスに「チェックマーク」を入れます。

音声合成処理を行うたびに表示内容が自動的に更新されます。

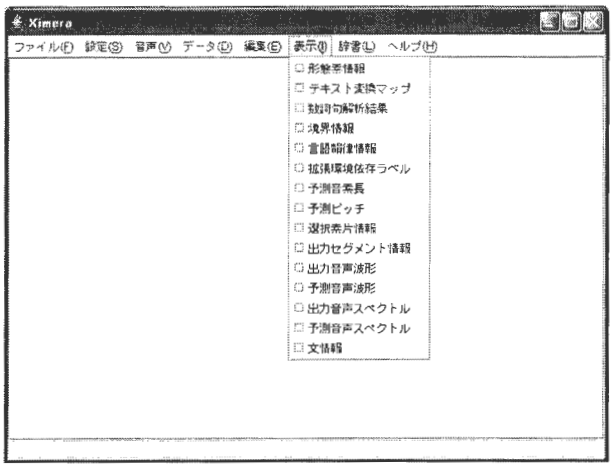


図 ２．４８ 中間データ表示選択設定画面

２．６．１ 形態素情報

茶筌で形態素解析されて出力された形態素情報です。

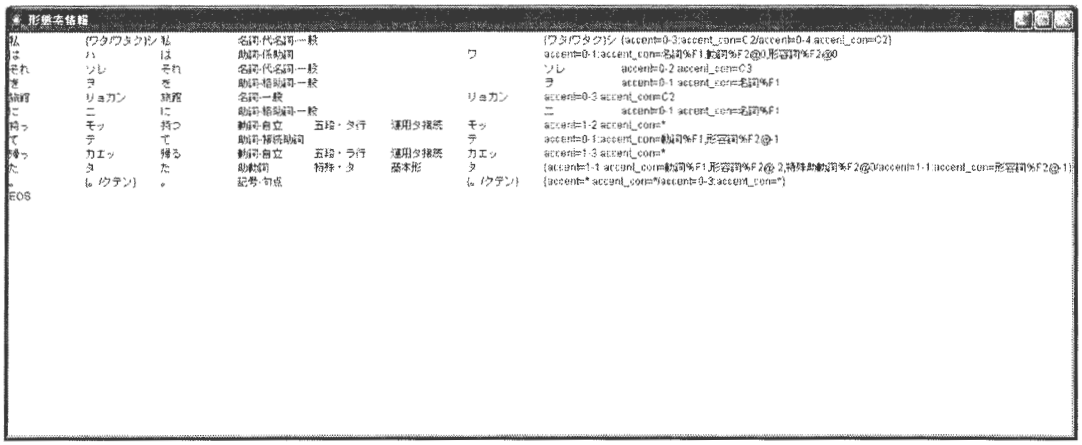


図 ２．４９ サブウィンドウ表示画面例 ＝形態素情報＝

２．６．２ テキスト変換マップ

形態素解析の前に、入力したテキストを形態素解析用テキストに変換した際の対応関係の情報です。

入力された数字を漢字に、半角文字→全角文字などの変換処理が行われます。

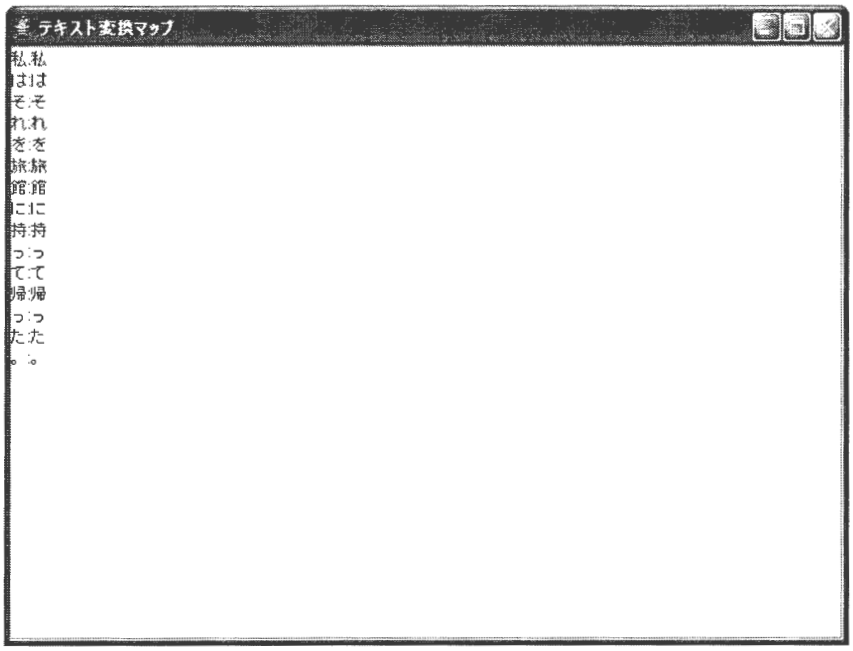


図 ２．５０ サブウィンドウ表示画面例 ＝テキスト変換マップ＝

２．６．３ 数詞句解析結果

形態素情報を入力として数詞句解析を行った結果です。

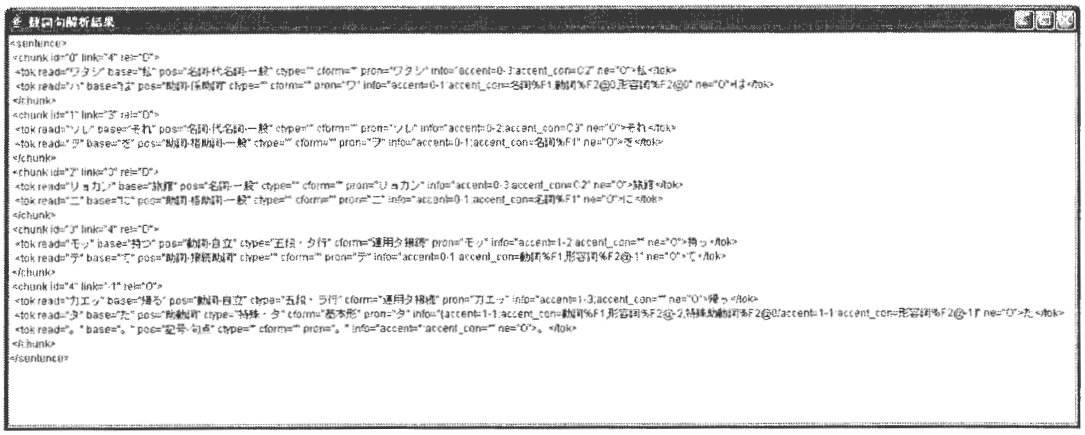


図 ２．５１ サブウィンドウ表示画面例 ＝数詞句解析結果＝

２．６．４ 境界情報

統語境界の解析を行った結果です。

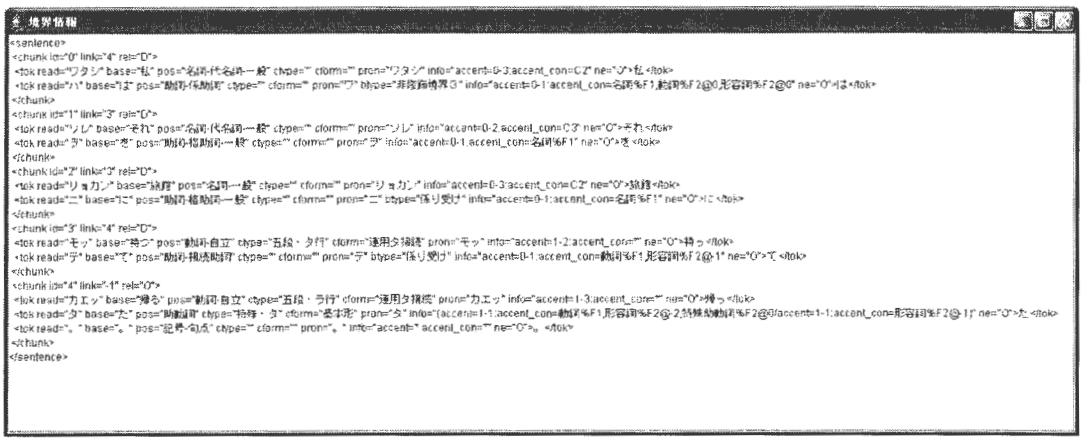


図 ２．５２ サブウィンドウ表示画面例 ＝境界情報＝

２．６．５ 言語韻律情報

形態素情報から生成された言語韻律情報です。

言語韻律情報

```
<?xml version="1.0" encoding="Shift_JIS"?>
<PLInfo>
<Head>
<Author Date="2004/6/23"> mor2pli</Author>
</Head>

<Body>

 <Sentence MorphSet="ipadic">
  <Morph Pos="名詞-代名詞-一般" Kanji="私">ワタシ</Morph>
  <Morph Pos="助詞-係助詞" Kanji="は" BoundaryType="非修飾境界３">ハ</Morph>
  <Morph Pos="名詞-代名詞-一般" Kanji="それ">ソレ</Morph>
  <Morph Pos="助詞-格助詞-一般" Kanji="を">ヲ</Morph>
  <Morph Pos="名詞-一般" Kanji="旅館">リョカン</Morph>
  <Morph Pos="助詞-格助詞-一般" Kanji="に" BoundaryType="係り受け">ニ</Morph>
  <Morph Pos="動詞-自立" Kanji="持っ" ConjugateType="五段・タ行" ConjugateForm="連用タ接続">モッ</Morph>
  <Morph Pos="助詞-接続助詞" Kanji="て" BoundaryType="係り受け">テ</Morph>
  <Morph Pos="動詞-自立" Kanji="帰っ" ConjugateType="五段・ラ行" ConjugateForm="連用タ接続">カエッ</Morph>
  <Morph Pos="助動詞" Kanji="た" ConjugateType="特殊・タ" ConjugateForm="基本形">タ</Morph>
 </Sentence>

 <Utterance>
  <ProsodicWord Accent="0" Pause="0.35">
   <Mora Kana="ワ">ワ</Mora>
   <Mora Kana="タ">タ</Mora>
   <Mora Kana="シ">シ</Mora>
   <Mora Kana="ハ">ワ</Mora>
  </ProsodicWord>
  <ProsodicWord Accent="0">
   <Mora Kana="ソ">ソ</Mora>
   <Mora Kana="レ">レ</Mora>
   <Mora Kana="ヲ">ヲ</Mora>
  </ProsodicWord>
  <ProsodicWord Accent="0">
   <Mora Kana="リョ">リョ</Mora>
   <Mora Kana="カ">カ</Mora>
   <Mora Kana="ン">ン</Mora>
   <Mora Kana="ニ">ニ</Mora>
  </ProsodicWord>
  <ProsodicWord Accent="1">
   <Mora Kana="モ">モ</Mora>
   <Mora Kana="ッ">ッ</Mora>
   <Mora Kana="テ">テ</Mora>
  </ProsodicWord>
  <ProsodicWord Accent="1">
   <Mora Kana="カ">カ</Mora>
   <Mora Kana="エ">エ</Mora>
   <Mora Kana="ッ">ッ</Mora>
   <Mora Kana="タ">タ</Mora>
  </ProsodicWord>
 </Utterance>

</Body>
</PLInfo>
```

図 ２．５３ サブウィンドウ表示画面例 ＝言語韻律情報＝

２．６．６ 拡張環境依存ラベル

言語韻律情報から生成された拡張環境依存ラベルです。

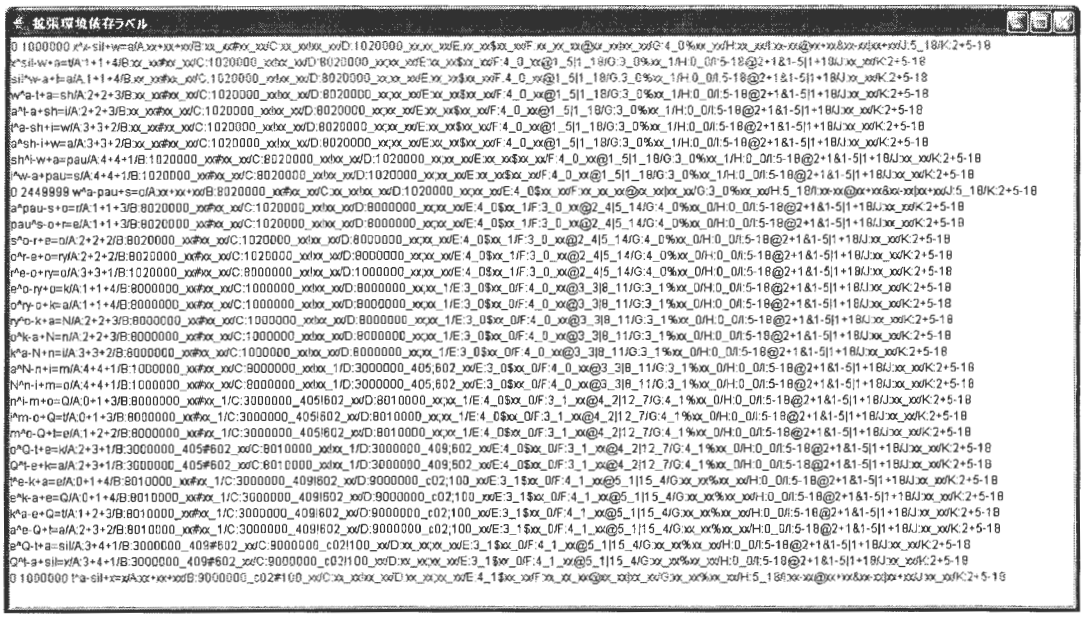


図 ２．５４ サブウィンドウ表示画面例 ＝拡張環境依存ラベル＝

２．６．７ 予測音素長

拡張環境依存ラベルから生成された予測音素長です。

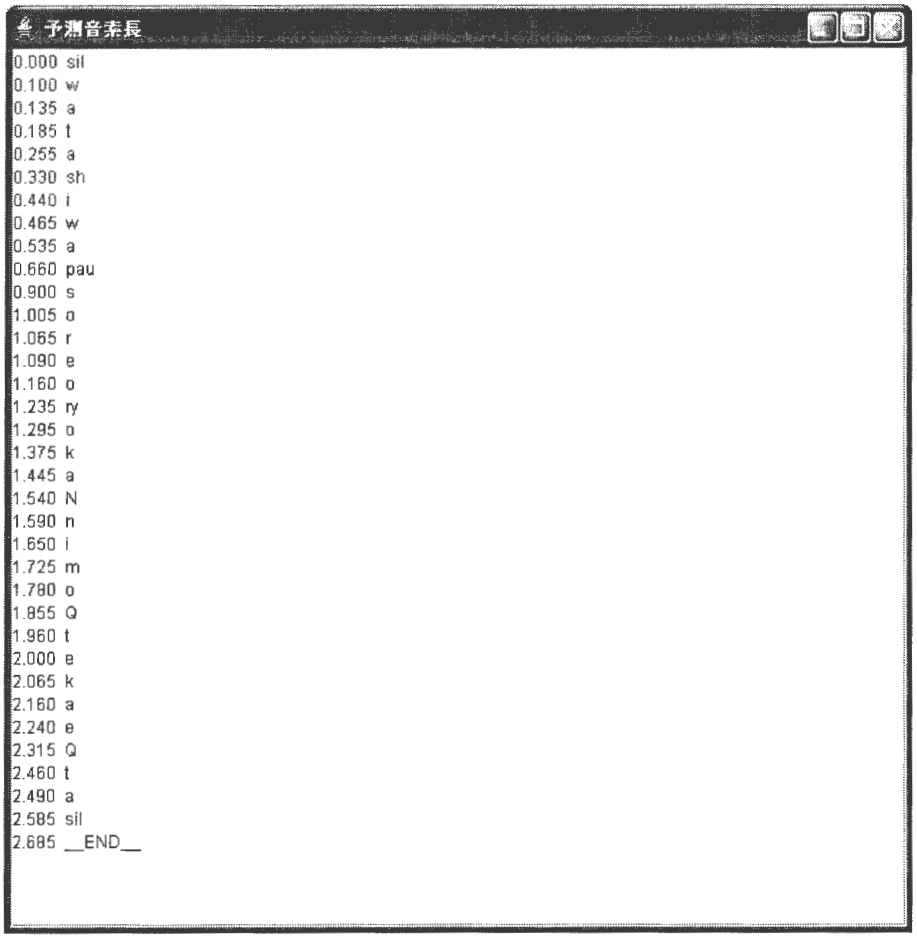


図 ２．５５ サブウィンドウ表示画面例 ＝予測音素長＝

２．６．８ 予測ピッチ

拡張環境依存ラベルから生成された予測ピッチです。

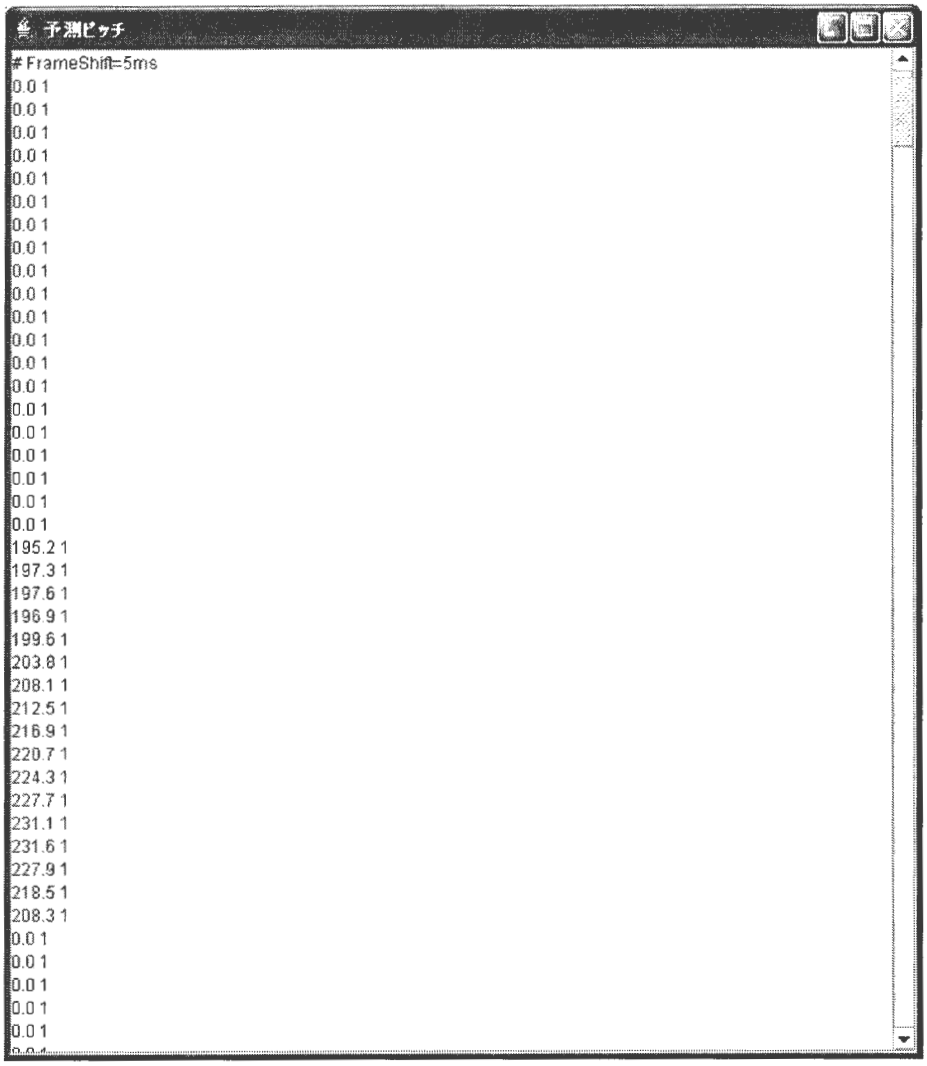


図 ２．５６ サブウィンドウ表示画面例 ＝予測ピッチ＝

２．６．９ 選択素片情報

音声素片選択処理の結果です。

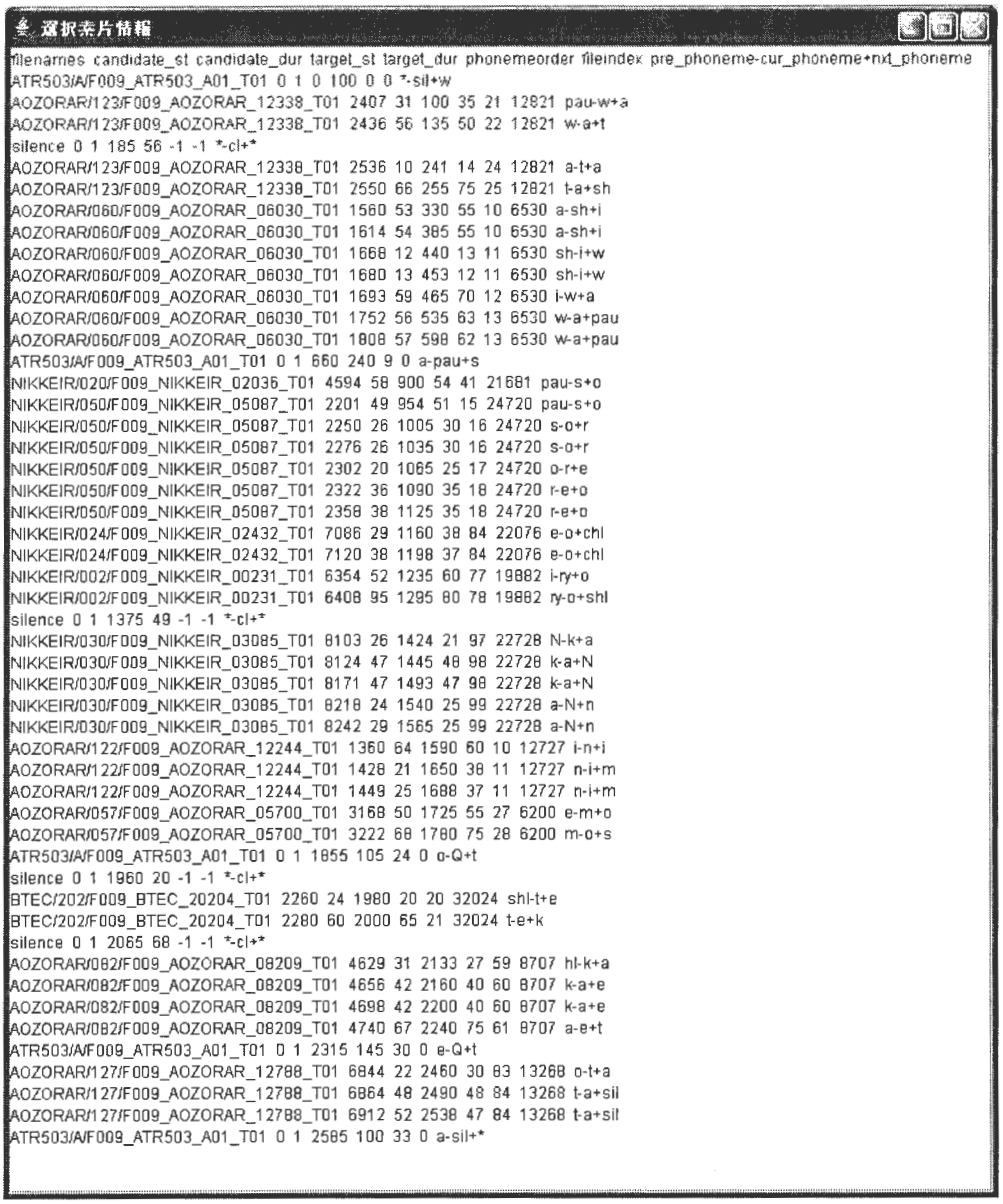


図 ２．５７ サブウィンドウ表示画面例 ＝選択素片情報＝

２．６．１０ 出力セグメント情報

合成音声として出力された音素と音素長です。

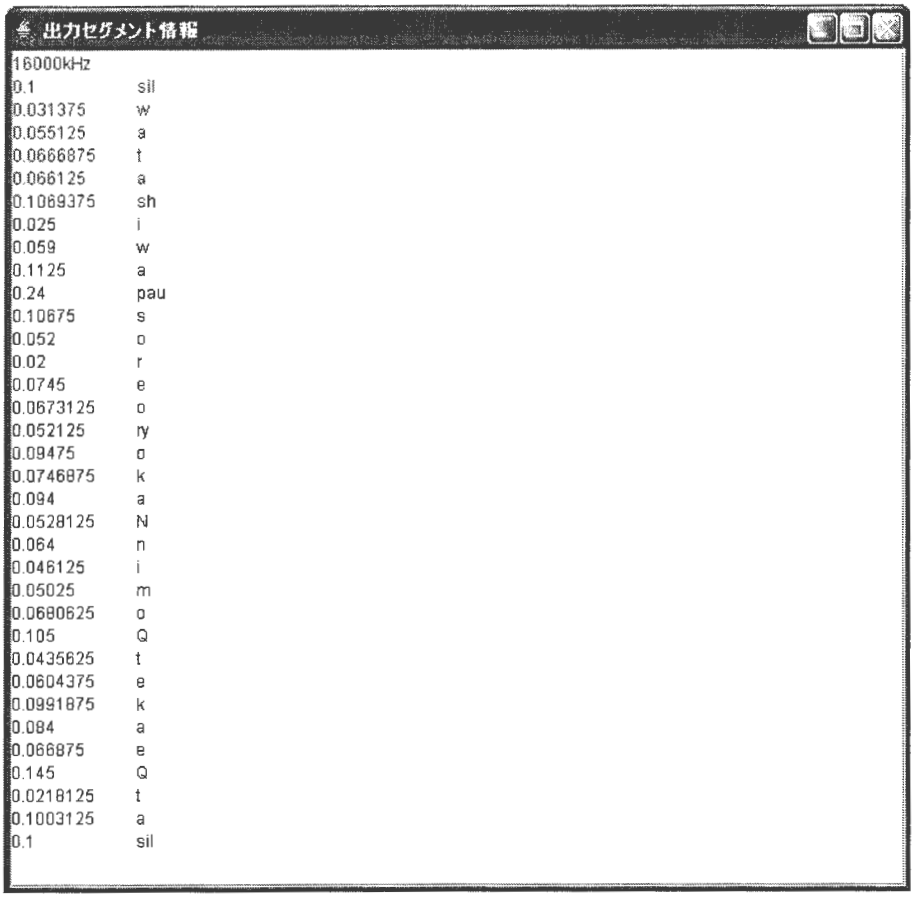


図 ２．５８ サブウィンドウ表示画面例 ＝出力セグメント情報＝

２．６．１１ 出力音声波形

出力された合成音声の音声波形です。

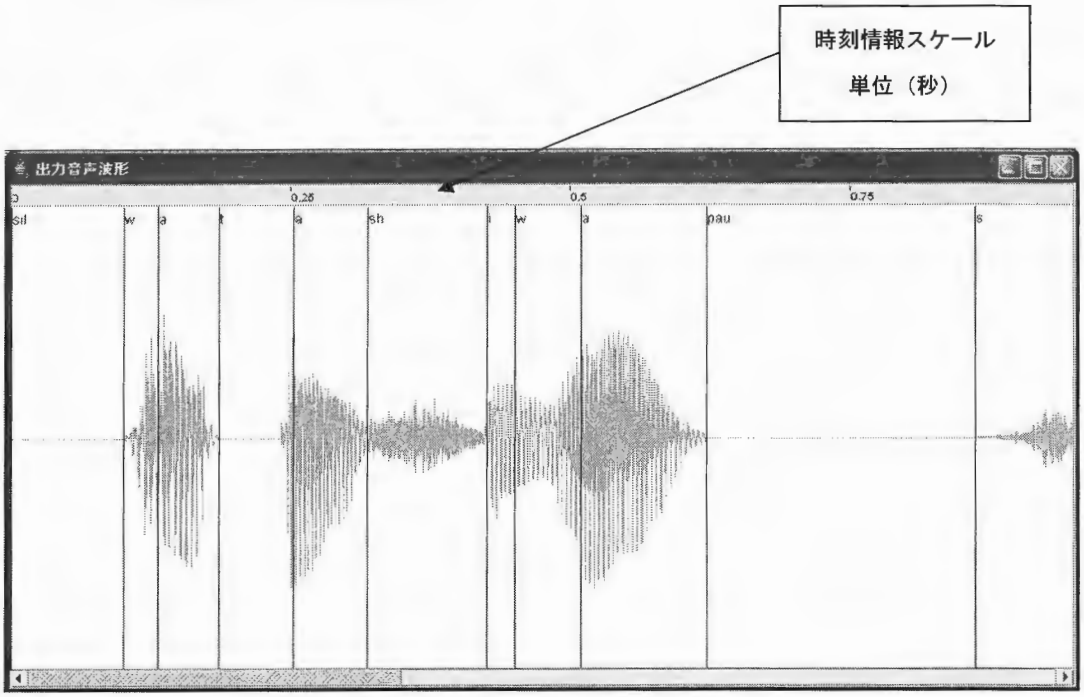


図 ２．５９ サブウィンドウ表示画面例 ＝出力音声波形＝

※出力音声波形サブウィンドウで、出力された合成音声を再生できます。
再生したい箇所をマウス左ボタンでドラッグして選択し、出力音声波形サブウィンドウ上の任意の場所で、マウス右ボタンでポップアップメニューを表示し［再生］を押下します。

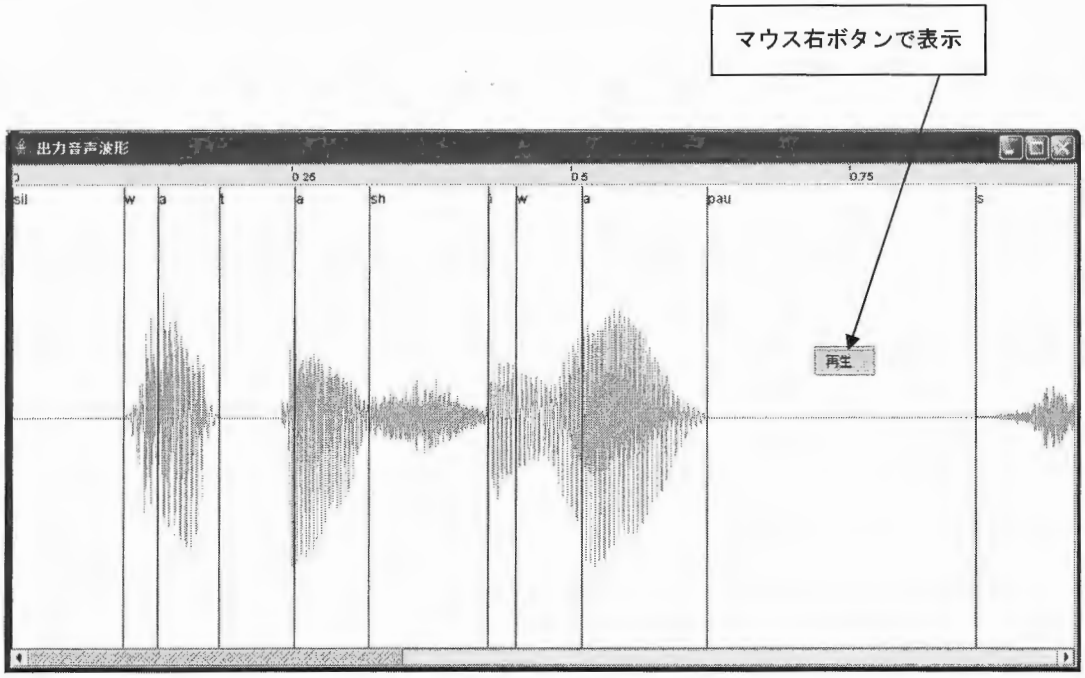


図 2.60 サブウィンドウ表示画面例 =合成音声再生時=

２．６．１２　HTS 合成音声波形（予測音声波形）

HTS の出力から生成した合成音声の音声波形です。

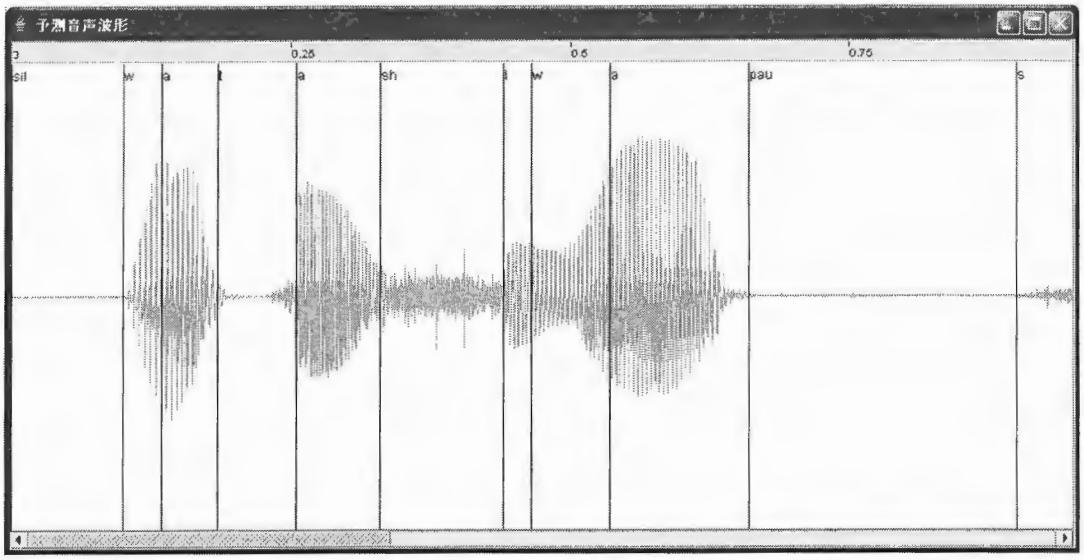


図 ２．６１ サブウィンドウ表示画面例　＝予測音声波形＝

※予測音声波形サブウィンドウで、HTS 予測結果から生成された合成音声を再生できます。再生したい箇所をマウス左ボタンでドラッグして選択し、予測音声波形サブウィンドウ上の任意の場所で、マウス右ボタンでポップアップメニューを表示し［再生］を押下します。

２．６．１３ 出力音声スペクトログラム

出力された合成音声のスペクトルです。

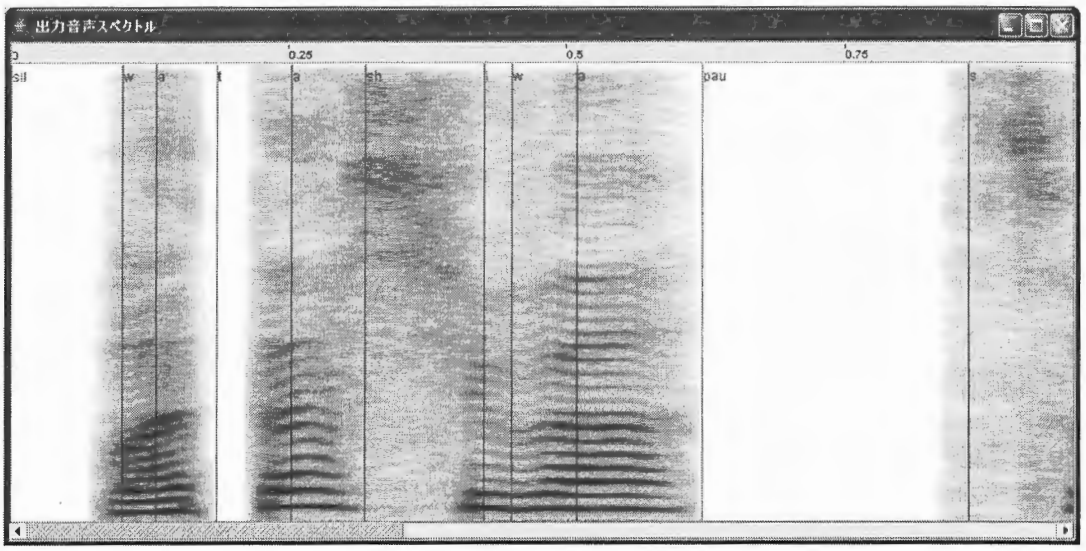


図 ２．６２ サブウィンドウ表示画面例 ＝出力音声スペクトル＝

※出力音声スペクトルサブウィンドウで、出力された合成音声を再生できます。
再生したい箇所をマウス左ボタンでドラッグして選択し、出力音声スペクトルサブウィンドウ上の任意の場所で、マウス右ボタンでポップアップメニューを表示し［再生］を押下します。

２．６．１４　HTS 合成音声スペクロトグラム（予測音声スペクトログラム）

HTS の出力から生成した合成音声のスペクトルです。

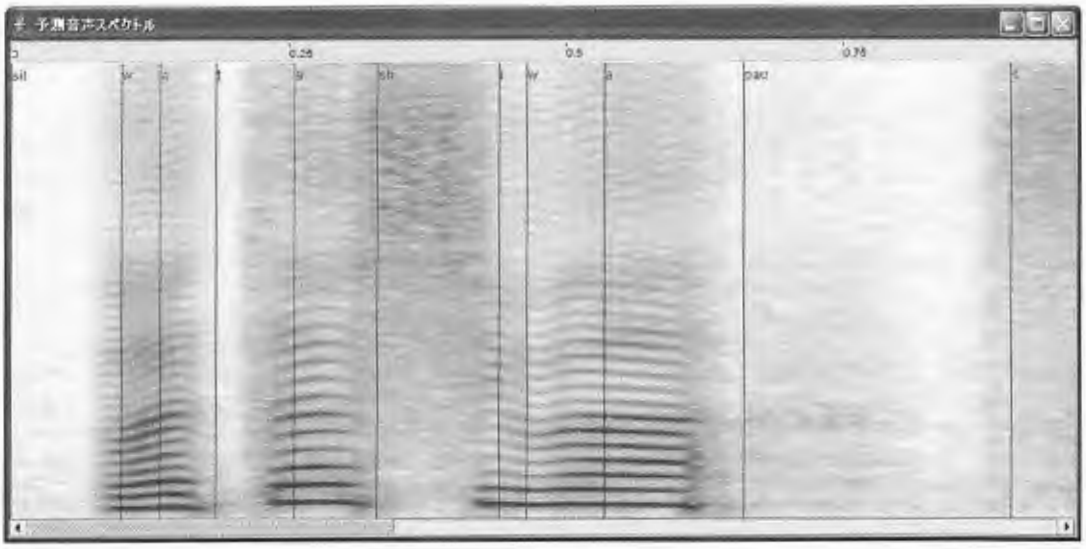


図 ２．６３ サブウィンドウ表示画面例　＝予測音声スペクトル＝

※予測音声スペクトルサブウィンドウで、HTS 予測結果から生成された合成音声を再生できます。
再生したい箇所をマウス左ボタンでドラッグして選択し、予測音声スペクトルサブウィンドウ上の任意の場所で、マウス右ボタンでポップアップメニューを表示し［再生］を押下します。

２．６．１５ 文情報

入力テキスト（単語）と予測結果として出力される音素との対応表です。

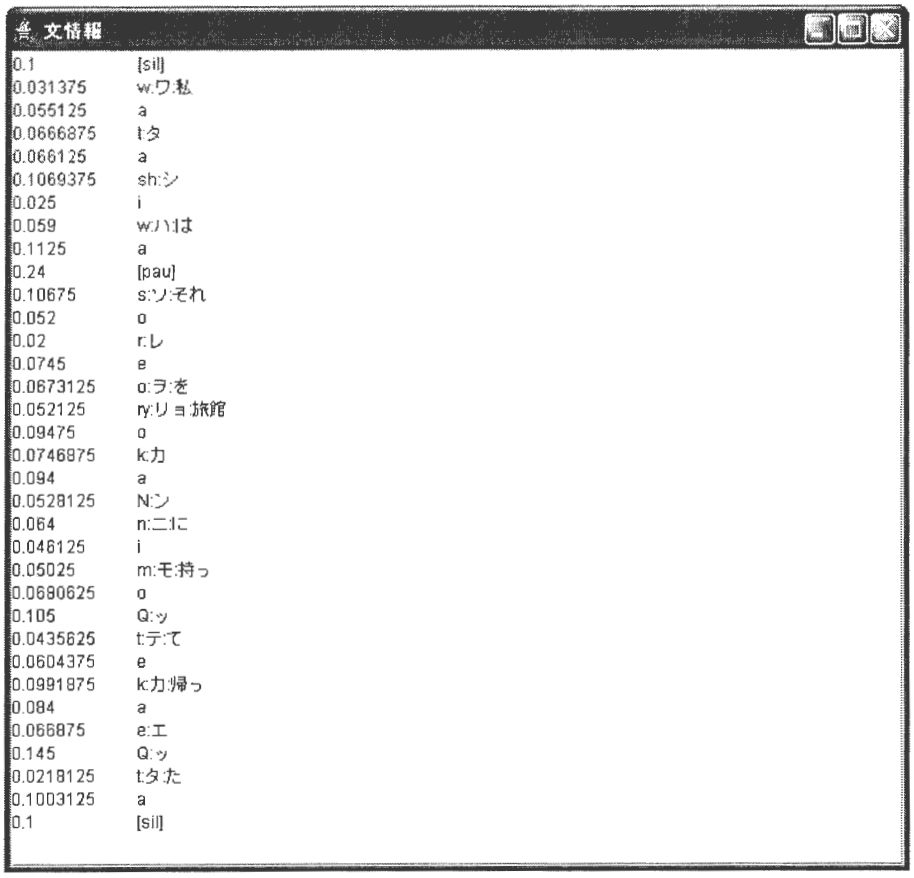


図 ２．６４ サブウィンドウ表示画面例 ＝文情報＝

２．７ 辞書

既存の辞書以外に、ユーザー辞書を使用できます。

２．７．１ ユーザー辞書への新規登録・追加登録（個別登録）

新しくユーザー辞書登録をする場合、XIMERA GUI メインウィンドウメニューバーの【辞書（L)】[辞書の追加] メニューを実行します。
辞書登録の入力画面から、辞書名（任意名）・表記・読みを入力し、品詞欄のプルダウンメニューから該当する品詞を選択します。
品詞の種類については＜表 2.4 辞書登録品詞一覧＞を参照してください。

＜例 1＞
「蒲公英の花が咲いています。」と入力した場合、XIMERA Version1.1 XIMERA GUIメイン辞書では、「ほこうえいのはながさいています。」と解析されます。
ここで、「ほこうえい」を「たんぽぽ」と解析されるよう辞書登録をします。

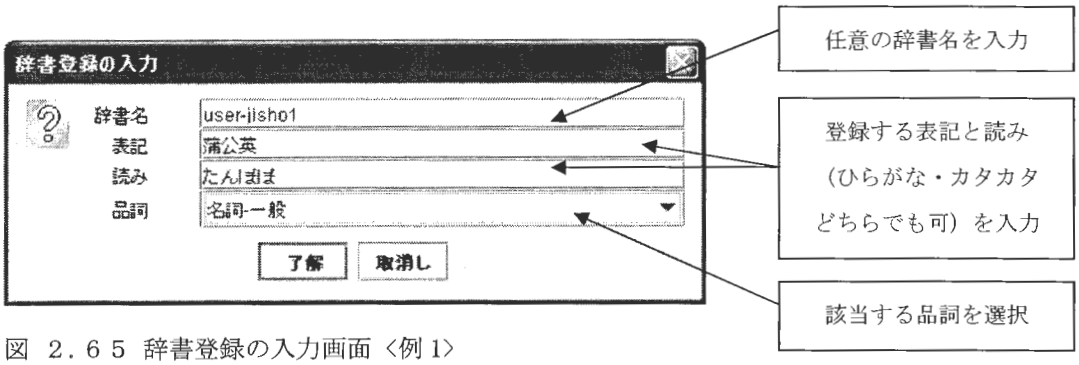


図 ２．６５ 辞書登録の入力画面 〈例 1〉

了解ボタンを押下した時点で、Ximera110¥xdata¥settings¥windows¥ipadic のディレクトリにユーザー辞書を使用するために必要なファイル（ user-jisho1.da、user-jisho1.dat、user-jisho1.dic、user-jisho1.lex ）が作成されます。
“user-jisho1.dic”の中に、「蒲公英（たんぽぽ）」が登録されます。

■Linux 版■
Ximera110/xdata/settings/linux/ipadic のディレクトリに作成されます。

```
(品詞 (名詞 一般)) ((見出し語 (蒲公英 1)) (読み たんぽぽ) (発音 たんぽぽ) )
```

図 2.66 user-jishol.dic のフォーマット

※1：違う単語（表記）を同じ辞書に追加登録する場合は、上記作業を繰り返します。

※2：別の辞書名で登録する場合は、辞書名を別の任意名に変えて入力します。

表 2.4 辞書登録品詞一覧

辞書登録時に指定する品詞の種類（全 73 種）			
名詞 一般	名詞 接尾 一般	動詞 接尾	助詞 副詞化
名詞 固有名詞 一般	名詞 接尾 人名	形容詞 自立	助詞 特殊
名詞 固有名詞 人名 一般	名詞 接尾 地域	形容詞 非自立	助動詞
名詞 固有名詞 人名 姓	名詞 接尾 サ変接続	形容詞 接尾	感動詞
名詞 固有名詞 人名 名	名詞 接尾 助動詞語幹	副詞 一般	記号 一般
名詞 固有名詞 組織	名詞 接尾 形容動詞語幹	副詞 助詞類接続	記号 句点
名詞 固有名詞 地域 一般	名詞 接尾 副詞可能	連体詞	記号 読点
名詞 固有名詞 地域 国	名詞 接尾 助数詞	接続詞	記号 空白
名詞 代名詞 一般	名詞 接尾 特殊	助詞 格助詞 一般	記号 アルファベット
名詞 代名詞 縮約	名詞 接続詞的	助詞 格助詞 引用	記号 括弧開
名詞 副詞可能	名詞 動詞非自立的	助詞 格助詞 連語	記号 括弧閉
名詞 サ変接続	名詞 引用文字列	助詞 接続助詞	その他 間投
名詞 形容動詞語幹	名詞 ナイ形容詞語幹	助詞 係助詞	フィラー
名詞 数	接頭詞 名詞接続	助詞 副助詞	非言語音
名詞 非自立 一般	接頭詞 動詞接続	助詞 間投助詞	語断片
名詞 非自立 副詞可能	接頭詞 形容詞接続	助詞 並立助詞	未知語
名詞 非自立 助動詞語幹	接頭詞 数接続	助詞 終助詞	
名詞 非自立 形容動詞語幹	動詞 自立	助詞 副助詞／並立助詞／終助詞	
名詞 特殊 助動詞語幹	動詞 非自立	助詞 連体化	

２．７．２ ユーザー辞書への新規登録・追加登録（一括登録）

あらかじめ準備したテキストファイルをロードして、ユーザー辞書を一括登録します。

ロードするテキストファイルのフォーマットは、

〔単語（表記）〕，〔読み〕，〔品詞〕の順に半角カンマまたはタブで区切ってください。

品詞の種類については＜表 2.4 辞書登録品詞一覧＞を参照してください。

品詞を指定しない場合は、デフォルトで〔名詞 一般〕が選択されます。

＜注意＞

〔品詞〕の中のさらに細かい分類は、半角スペースで区切ってください。

〔名詞　固有名詞　地域　国〕

```
#固有名詞<1>

西班牙,スペイン,名詞 固有名詞 地域 国
洪牙利,ハンガリー,名詞 固有名詞 地域 国
墺太利,オーストリア,名詞 固有名詞 地域 国
濠太剌利,オーストラリア,名詞 固有名詞 地域 国

;固有名詞<2>

布哇,ハワイ,名詞 固有名詞 地域 一般
```

＜注意＞
； 改行 は
コメントとして無視
されます。

図 ２．６７ ユーザー辞書一括登録用テキストファイル（.txt）フォーマット例

XIMERA GUI メインウィンドウメニューバーの【辞書（L)】[辞書の一括登録] メニューを実行します。

表示された読み込みファイルの選択画面でテキストファイルを選択します。

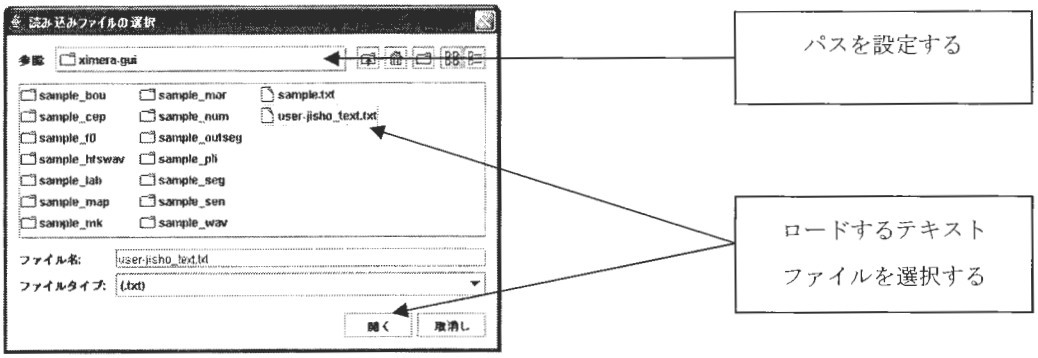


図 ２．６８ 読み込みファイルの選択画面

辞書名の入力画面から、辞書名（任意名）を入力します。

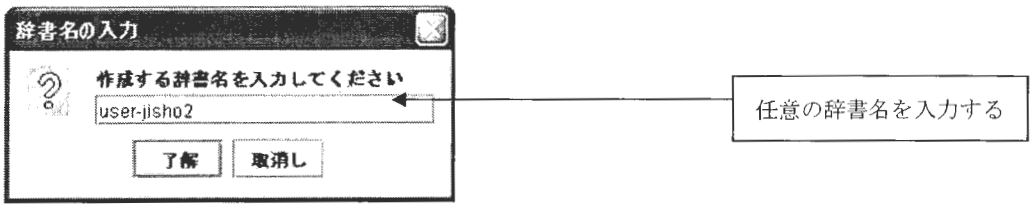


図 2.69 辞書名入力画面

了解ボタンを押下した時点で、Ximera110¥xdata¥settings¥windows¥ipadic のディレクトリにユーザー辞書を使用するために必要なファイル（ user-jisho2.da、user-jisho2.dat、user-jisho2.dic、user-jisho2.lex ）が作成されます。

“user-jisho2.dic”の中に、

「西班牙（スペイン）」
「洪牙利（ハンガリー）」
「墺太利（オーストリア）」
「濠太剌利（オーストラリア）」
「布哇（ハワイ）」　　が登録されます。

■Linux 版■

Ximera110/xdata/settings/linux/ipadic のディレクトリに作成されます。

```
(品詞 (名詞 固有名詞 地域 国)) ((見出し語 (西班牙 1)) (読み スペイン) (発音 スペイン) )
(品詞 (名詞 固有名詞 地域 国)) ((見出し語 (洪牙利 1)) (読み ハンガリー) (発音 ハンガリー) )
(品詞 (名詞 固有名詞 地域 国)) ((見出し語 (墺太利 1)) (読み オーストリア) (発音 オーストリア) )
(品詞 (名詞 固有名詞 地域 国)) ((見出し語 (濠太剌利 1)) (読み オーストラリア) (発音 オーストラリア) )
(品詞 (名詞 固有名詞 地域 一般)) ((見出し語 (布哇 1)) (読み ハワイ) (発音 ハワイ) )
```

図 2.70 user-jisho2.dic のフォーマット

※1：他のテキストファイルをロードして既存のユーザー辞書に追加する場合は、辞書名の入力画面で既存のユーザー辞書名を入力します。

※2：別の辞書名で登録する場合は、辞書名を別の任意名に変えて入力します。

＜注意＞

ロードするファイルは、文字コードが Windows 版：SJIS、Linux 版：EUC で記述されていることをご確認ください。

２．７．３ 辞書のロード

登録した辞書でテキストを解析するためには、辞書をロードする必要があります。

XIMERA GUI メインウィンドウメニューバーの【辞書（L)】[辞書のロード] メニューを実行します。

辞書名の指定画面で、ロードする辞書名を指定します。

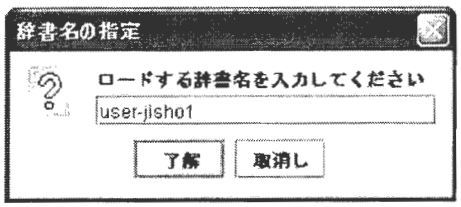


図 ２．７１ 辞書名の指定画面

P.55 ＜例１＞で示した「蒲公英の花が咲いています。」という文を XIMERA GUI メインウィンドウのテキスト入力エリアに入力して、メニューバーの【音声（V)】[音声の合成と再生（Ctrl-P）] メニューを実行します。

「たんぽぽのはながさいています。」と解析されて合成音声が生成され、再生されます。

２．７．４ 辞書のアンロード

ユーザー辞書をアンロードします。

XIMERA GUI メインウィンドウメニューバーの【辞書（L)】[辞書のアンロード] メニューを実行します。

辞書名の指定画面で、アンロードする辞書名を指定します。

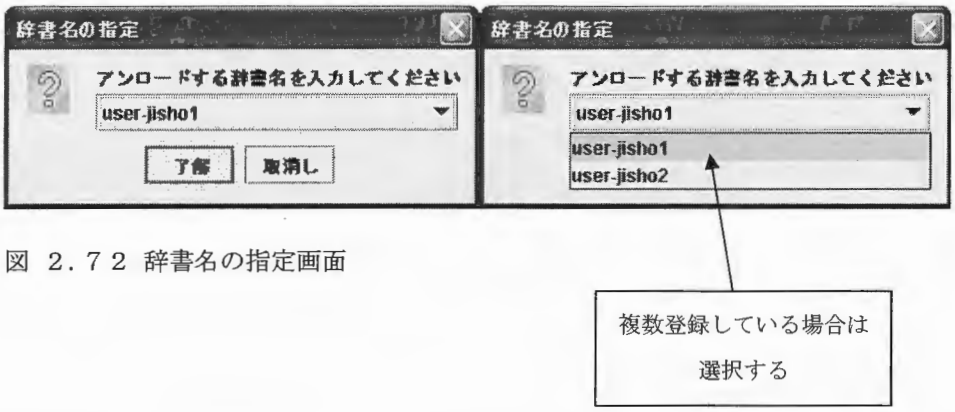


図 ２．７２ 辞書名の指定画面

※ユーザー辞書を登録・ロードしていない場合は、【辞書（L)】[辞書のアンロード] メニューを実行しても辞書名の指定画面は表示されません。

２．７．５ 辞書の消去

ユーザー辞書を消去します。

XIMERA GUI メインウィンドウメニューバーの【辞書（L)】[辞書の消去] メニューを実行します。

辞書名の指定画面で、消去する辞書名を指定します。

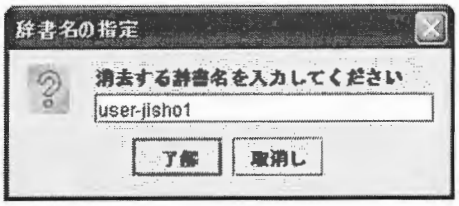


図 ２．７３ 辞書名の指定画面

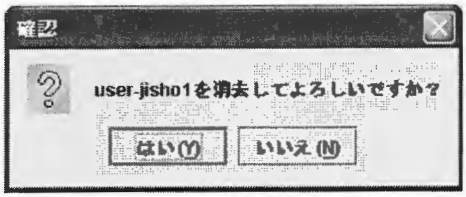


図 ２．７４ 消去確認画面

※辞書を消去すると、Ximera110¥xdata¥settings¥windows¥ipadic のディレクトリにある（ user-jisho1.da、user-jisho1.dat、user-jisho1.dic、user-jisho1.lex ）が削除されるため、辞書の復活はできません。ご注意ください。

■Linux 版■

Ximera110/xdata/settings/linux/ipadic のディレクトリにある（ user-jisho1.da、user-jisho1.dat、user-jisho1.dic、user-jisho1.lex ）が削除されます。

２．８ ヘルプ

バージョン情報のみの記載となっています。

XIMERA GUI メインウィンドウメニューバーの【ヘルプ (H)】[バージョン情報 (Ctrl-Y)] メニューを実行します。

3．その他のサンプルプログラム（頻度情報作成ツール）

頻度情報作成ツール

このツールは、テスト文をあらかじめ音声合成し、その素片選択結果より、コーパス中の素片データの使用頻度を求めるツールです。この使用頻度を利用することで、音声合成に使用されるコーパス中の音声ファイルの素片データを先読みし、高速化に利用することができます。高速化する処理は主に音声ファイルの open と read です。

まず、テスト文より素片選択結果を作成します。作成方法はサンプルプログラム xibatsu を変更し、素片選択結果を出力するプログラムを作成します。内容については、後述のサンプルプログラムを参照してください。

出力された素片選択結果ファイル（*.seg）を入力データとし、頻度情報作成ツール（segfreq）を実行します。

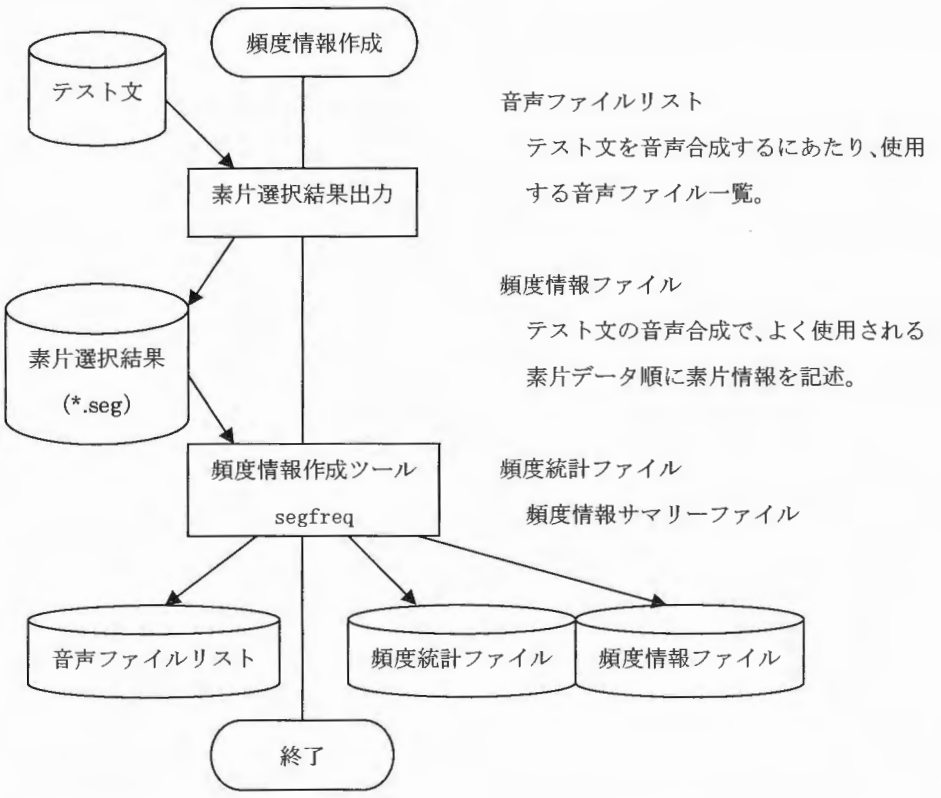


図 3．1 頻度情報作成ツール処理の流れ

頻度情報の利用

作成した頻度情報ファイル、音声ファイルリストをデータベースのディレクトリへコピーし、data_wvc.txt ファイルへオプションの記述を追加します。(図 3.2 網掛け部分)

追加後、XIMERA GUI を起動し、オプションを変更したデータベースを選択し、ロードしてください。-wavcachefreq の数字を大きくするとデータベースのロードに時間を要します。

＜注意＞このオプションを使用すると、データベースのロードに時間がかかり、SAPI で使用した場合、タイムアウトする可能性があります。

data_wvc.txt の内容

```
-f 16000
-shift 5.0
-f0shift 10.0
-wavdir ../../corpus/16k
-wavsign wav
-wavcacheidx fidx_50k.txt
-wavcacheseg freq_50k.txt
-wavcachefreq 50.0
```

左図の例のように

-wavcacheidx に音声ファイルリストのファイル名を記述します。

-wavcacheseg に頻度情報ファイル名を記述します。

-wavcachefreq に延べセグメント総数に対する使用率（単位%）を記述します。

例では

fidx_50k.txt：音声ファイルリストファイル名

freq_50k.txt：頻度情報ファイル名

が記述されています。

頻度統計ファイル（summary_50k.txt）の内容

```
セグメント総数(異なり):294077
セグメント総数(延べ)  :4854451
被覆率対セグメント数
     10.00      729 (0.25)
     20.00     2493 (0.85)
     30.00     5489 (1.87)
     40.00    10182 (3.46)
     50.00    17237 (5.86)
     60.00    27784 (9.45)
     70.00    43861 (14.91)
     80.00    69906 (23.77)
     90.00   118491 (40.29)
     95.00   166909 (56.76)
     99.00   247602 (84.20)
```

延べセグメント総数に対する使用率

異なるセグメント数

異なるセグメント総数に対する割合（単位%）

図 3.2 data_wvc.txt の内容 summary_50k.txt の内容

頻度情報ファイル作成手順

(1) 素片選択結果の作成

テスト文を入力データとし、後述のサンプルプログラムを実行します。

例として、Windows 環境で動作させた場合を記述します。また、説明のため、下記構成により実行されたものとして説明します。

実行ディレクトリ：	D:¥segtest
出力ディレクトリ先：	実行位置の直下
segdata ディレクトリ作成：	D:¥segtest¥segdata
データベースインストール先：	D:¥Ximera110¥xdata
素片選択結果出力プログラム名：	outseg
テスト文：	test.txt
話者：	F009
データベース：	F009
HMM 学習結果：	F009_hts

実行コマンド

```
cd  D:¥segtest
outseg  D:¥Ximera110¥xdata  F009  F009_hts  F009  test.txt  segdata
```

```
あらゆる現実をすべて自分の方へねじ曲げたのだ。
都会では、出会う人のほとんどが見知らぬ人である。
（省略）
・
・
```

図 3.3 test.txt の内容

※テスト文は 1 文 1 行のフォーマットで記述します。

出力は、segdata ディレクトリへ F009_F009_hts_F009_????.seg ファイルが 1 文 1 ファイルで作成されます。

(2) 頻度情報ファイルの作成

実行コマンド

```
segfreq  segdata  seg  10K
```

segdata : (1) で出力されたディレクトリを指定

seg : (1) で出力されたファイルの拡張子

10K : 頻度情報ファイルのプレフィックス名

この場合、出力されるファイルは以下のファイル名で出力されます。

fidx_10k.txt	音声ファイル名一覧
freq_10k.txt	頻度情報
summary_10k.txt	頻度統計

サンプルプログラム

```
int main( int argc, char** argv )
{
     if( argc != 7) {
          cout << "usage: segdata datapath speaker hts seg txtfile outdir" << endl ;
          cout << "sample: segdata /home/xdata F009 F009_hts F009 batch.txt ./test503" << endl ;
          return 0 ;
     }

     string path = argv[ 1 ];
     string speaker = argv[ 2 ];
     string hts = argv[ 3 ];
     string seg = argv[ 4 ];
     string txtfile = argv[ 5 ];
     string outdir = argv[ 6 ];
     char wk_buff[512];

     try {
          //初期処理
          ximera_init( path.c_str() ) ;
          //話者ロード (DBロード)
          ximera_speaker( speaker.c_str(), hts.c_str(), seg.c_str() ) ;

          ifstream ifs( txtfile.c_str() ) ;
          int n = 0 ;

          while( ifs.good( ) ) {

               enum { bufsiz = 1000 } ;
               char buf[ bufsiz ] ;
               ifs.getline( buf, bufsiz ) ;
               if( ifs.eof( ) )
                    break ;

               char segfile[ 1000 ] ;
               sprintf( segfile, "%s/%s_%s_%s_%04d ", outdir.c_str(),
                                                      speaker.c_str(),
                                                      hts.c_str(),
                                                      seg.c_str(),
                                                      n ) ;
               cout << n << " " << buf << endl ;

               string w = segfile ;

               try {
                    //テキスト合成開始
                    ximera_text_to_morphs( buf ) ;
                    ximera_morphs_to_plinfo( ) ;
                    ximera_plinfo_to_labels( );
                    ximera_labels_to_targets( ) ;
                    ximera_synth_select( );
                    ximera_synth_connect( wavefile ) ;
                    //テキスト合成終了
                    //素片選択結果出力
                    ximera_save_select_seg     ( ( w + ".seg" ).c_str( ) ) ;
               }

               catch( char* s ) {
                    cerr << "FATAL ERROR: " << s << endl ;
                    ofstream ofs( ( w + ".error" ).c_str( ) ) ;
```

```
                    ofs << s << endl ;
                }

                catch( ... ) {
                    cerr << "UNKNOWN ERROR" << endl ;
                    ofstream ofs( ( w + ".error" ).c_str( ) ) ;
                    ofs << "unknown error" << endl ;
                }

                n ++ ;
            }
            ximera_cleanup( ) ;
        }

        catch( char* s ) {
            cerr << "FATAL ERROR: " << s << endl ;
            return 1 ;
        }

        catch( ... ) {
            cerr << "unknown error caught in main( )" << endl ;
            return 1 ;
        }

        return 0 ;
}
```

図 3.4 サンプルプログラム例

XIMERA（Ver1.1）

ユーザーズマニュアル

= データベース作成 =

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

本マニュアルは、ATR であらたに開発されたコーパスベース方式による音声波形素片接続型テキスト音声合成エンジン（開発コード名：XIMERA、以下「XIMERA」と記す）の、XIMERA エンジンで使用するデータベース、HMM の作成方法について説明します。読み上げ原稿を準備し、音声収録を実施した後に、本マニュアルの手順に従ってデータベース、HMM を作成してください。

以下のツールについて、使用方法を説明します。

- 音声切り出しツール（JSPSEG）
- サンプリングレート変換ツール（srconv）
- データベース作成 GUI（AutoDBmake）
- F0 ラベル修正ツール（PIIKUN）
- HTS 学習 GUI（HMM）

1． データベース作成フロー

データベースを作成するには、以下の作業を行います。

① コーパスの作成

収録音声と読み上げ原稿より、1 文単位の音声ファイル、言語韻律情報ファイル、カナファイルを作成します。

② データベースの作成

①で作成した音声ファイルとカナファイルからデータベースを作成します。

③ F0・ラベル修正

データベース作成 GUI で作成した F0 ファイル、ラベルファイルを必要に応じて編集します。

④ HMM の作成

①で作成した音声ファイル、言語韻律情報ファイルを元に HTS 学習を行い、HMM を作成します。

１．１ コーパスの作成

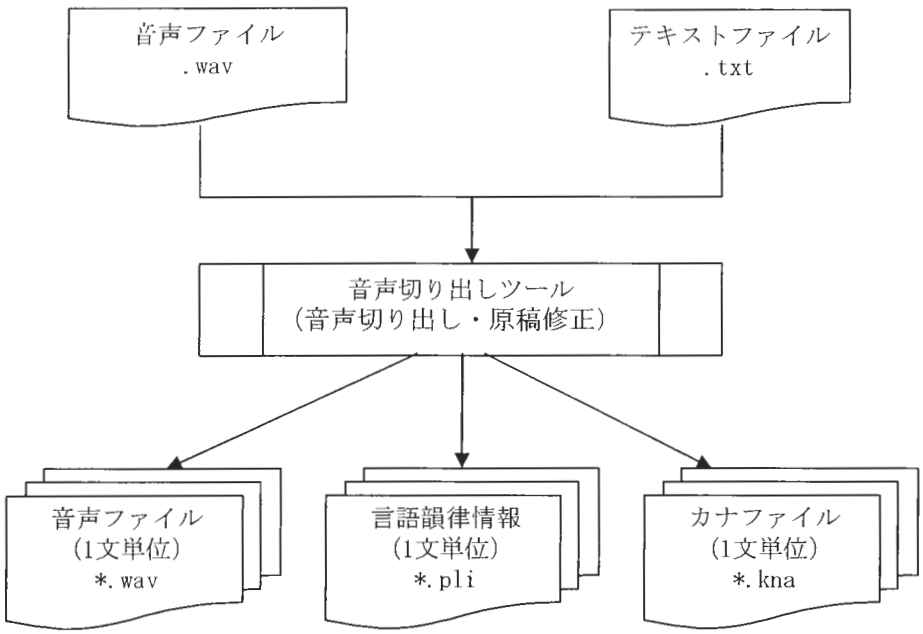


図 １．１ 音声コーパス作成フロー

＜説明＞

- 収録した音声ファイルを１文単位に分割し、音声ファイル、言語韻律情報ファイル、カナファイルを出力します。
- データベース作成には、音声切り出しにより作成したファイルのうち、音声ファイルとカナファイルが必要です。（言語韻律情報ファイルは不要です。）
- HTS 学習には、音声切り出しにより作成したファイルのうち、音声ファイルと言語韻律情報ファイルが必要です。（カナファイルは不要です。）
- HTS 学習の入力ファイルとなるカナファイルは、言語韻律情報を修正した後、言語韻律情報から作成します。（言語韻律情報と、カナファイルから作成したラベルファイルに不整合があると、HTS 学習に失敗するためです。）詳細は、「1.4 IMM 作成フロー」をご参照ください。

１．２ データベースの作成

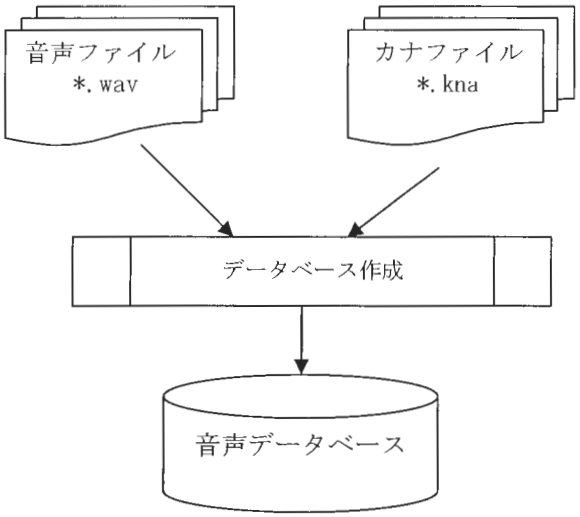


図 １．２ データベース作成フロー

＜説明＞

- データベース作成 GUI を使用してデータベースを作成します。
  詳細は、「5. データベース作成手順」をご参照ください。
- F0、ラベルを修正する場合はデータベース作成のラベル作成（工程名 align)、F0 抽出（工程名 f0） の工程が終了した後、F0・ラベル修正ツールで修正します。
  その後、修正結果を入力ファイルとしてデータベース作成を行います。

1．3 F0・ラベル修正フロー

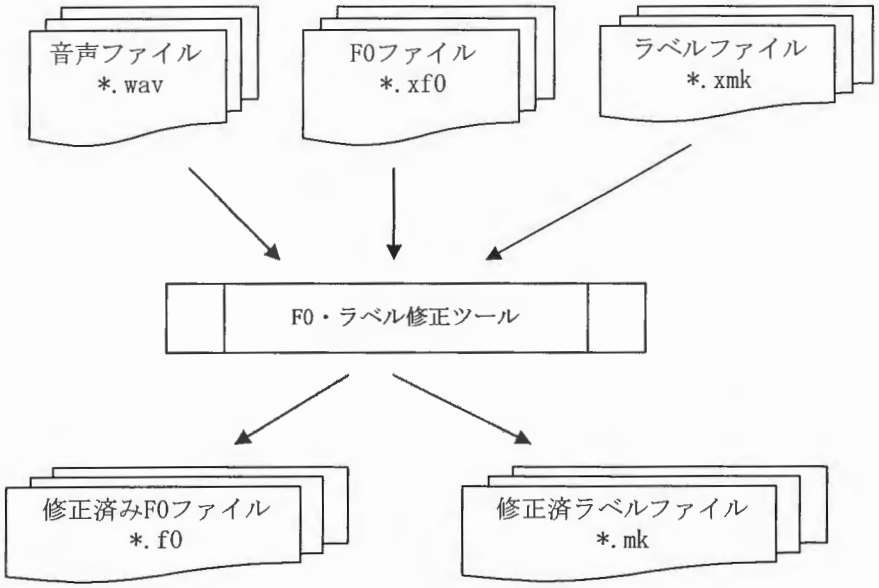


図 1．3 F0・ラベル修正フロー

＜説明＞

- データベース作成 GUI より作成した F0 ファイル(*.xf0)、ラベルファイル(*.xmk)を元に修正します。
- F0、ラベルの修正は必須ではありません。
- コーパスの一部のみ修正してもかまいません。
- 修正した F0、ラベル情報は、データベース作成 GUI で units 以降の工程を実行することによりデータベースに反映されます。
  詳細は、「5.データベース作成手順」をご参照ください。

１．４ HMM 作成フロー

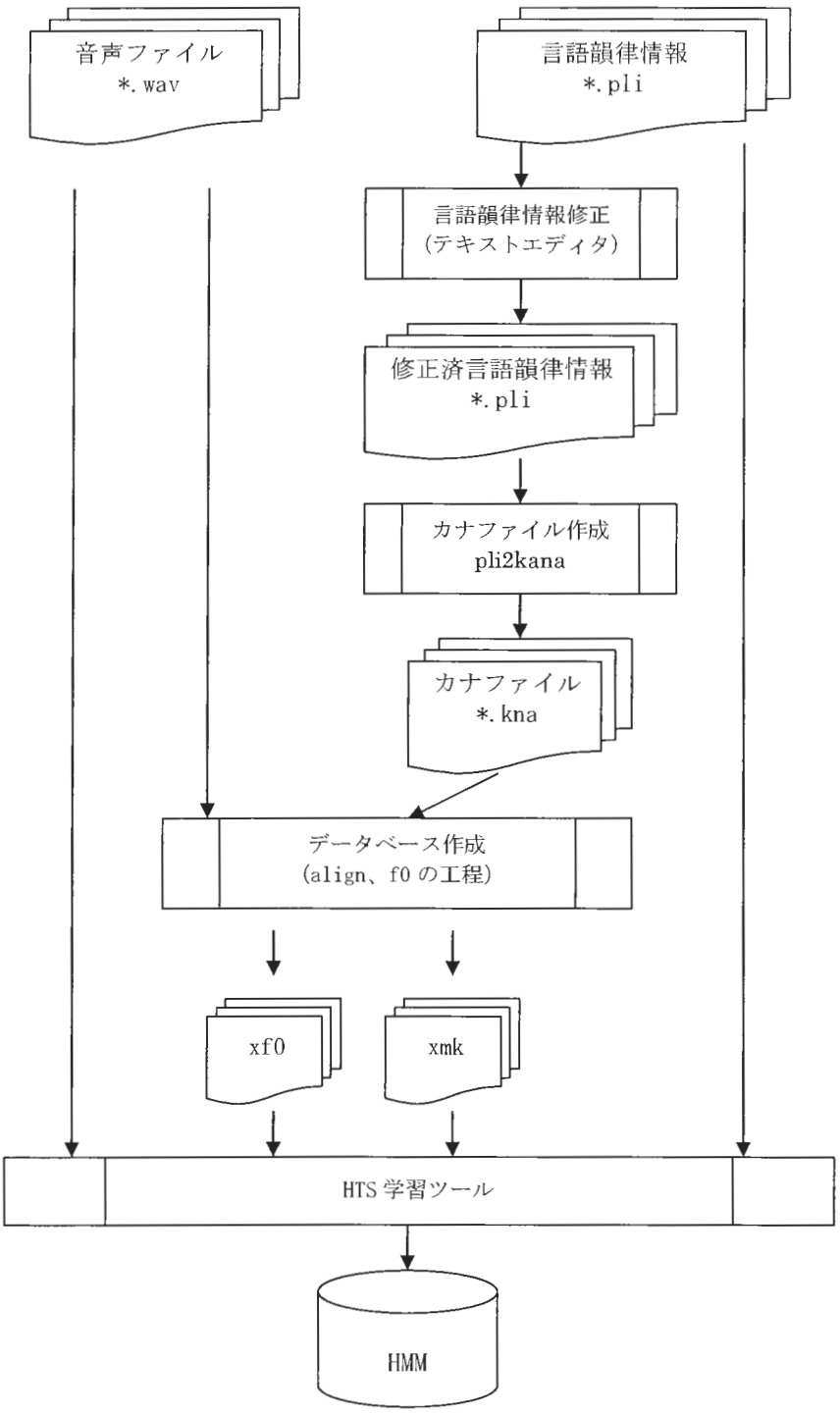


図 １．４ HMM 作成フロー

＜説明＞

- HTS 学習に必要な入力ファイルは、音声ファイル、言語韻律情報ファイル、F0 ファイル、ラベルファイルです。
- 言語韻律情報の修正は、お手持ちのテキストエディタ等をご使用ください。
- 読み付与間違いがある場合は、言語韻律情報ファイルを正しく修正してください。
- ツール pli2kana で言語韻律情報からカナファイルを作成し、データベース作成 GUI でラベルファイルを作成します。
- F0 ファイルとラベルファイルは、F0・ラベル修正ツールを使用して修正したものをご使用いただいてもかまいません。その場合はファイルの拡張子がそれぞれ、f0、 mk になり、xf0、xmk ファイルより優先的に参照します。

2． ディレクトリ構成

コーパス、音声データベース、HTS 学習結果のファイルは以下を参考に配置してください。

```
xdata
    ├F009（話者名）
    |    ├──corpus（コーパス）
    |    |    ├──08k（サンプリングレート名）
    |    |    |    ├──ATR503（コーパス名）
    |    |    |    |    ├──A
    |    |    |    |    |    └──A01.wav（8kHz）
    |    |    |    |    ├──B
    |    |    |    |    └──C
    |    |    |    ├──NIKKEI
    |    |    |    └──MAINICHIR
    |    |    ├──16k
    |    |    |    ├──ATR503
    |    |    |    |    ├──A
    |    |    |    |    |    ├──A01.wav（16kHz 音声ファイル）
    |    |    |    |    |    ├──A01.kna（カナファイル）
    |    |    |    |    |    ├──A01.pli（言語韻律情報ファイル HTS 学習にのみ必要）
    |    |    |    |    |    ├──A01.f0 （修正済み F0 ファイル 任意）
    |    |    |    |    |    ├──A01.xf0（F0 ファイル）
    |    |    |    |    |    ├──A01.mk （修正済みラベルファイル 任意）
    |    |    |    |    |    ├──A01.xmk（ラベルファイル）
    |    |    |    |    |    ├──A01.mcep（ケプストラムファイル）
    |    |    |    |    |    └──A01.pow（パワーファイル）
    |    |    |    |    ├──B
    |    |    |    |    └──C
    |    |    |    ├──NIKKEI
    |    |    |    └──MAINICHIR
    |    |    ├──48k
    |    |    |    ├──ATR503
    |    |    |    |    ├──A
    |    |    |    |    |    └──A01.wav（48kHz 音声ファイル）
    |    |    |    |    ├──B
    |    |    |    |    └──C
    |    |    |    ├──NIKKEI
    |    |    |    └──MAINICHIR
    |    |    |
    |    |    └──任意のサンプリングレート
    |    |
    |    ├──seg（データベース）
    |    |    ├──F009（データベース名）
    |    |    |    ├──data.txt（データベースファイルリスト）
    |    |    |    ├──data_wvc.txt （wav ディレクトリ指定等）
    |    |    |    ├──list.txt（音声ファイルリスト）
    |    |    |    └──info.txt（DB 作成時の情報 選択されたコーパス情報等）
    |    |    ├──F009_503
    |    |    └──任意のデータベース名
    |    |
    |    └──hts（HTS 学習結果）
```

```
|           ├─F009_hts（モデル名）
|           |     ├─mcep.pdf
|           |     └─info.txt（HMM 作成時の情報）
|           └─任意のモデル名
|
├任意の話者名（話者名）
|
└─settings
      ├─linux
      |     ├─ipadic
      |     |     ├─chadic.dat（辞書）
      |     |     └─udic.dic（ユーザー辞書 任意）
      |     |
      |     ├─ipadic.pos（品詞ファイル）
      |     ├─cabocharc（南瓜設定ファイル）
      |     └─chasenrc（茶筌設定ファイル）
      |
      └─windows
            ├─ipadic
            |     ├─chadic.dat（辞書）
            |     ├─sapi.dic（SAPI 用辞書）
            |     └─udic.dic（ユーザー辞書 任意）
            └─chasenrc
```

＜ディレクトリ構成説明＞

- 赤字で表示されたディレクトリ名は固定の名前です。変更しないでください。
- 青字で表示されたファイルはユーザーがツール等で作成して明示的に配置するファイルです。
- xdata ディレクトリ以下に、話者毎のディレクトリと settings ディレクトリがあります。
- 各話者ディレクトリに、corpus(コーパス)・seg(データベース)・hts(HTS 学習結果) ディレクトリが存在するようにしてください。
- corpus ディレクトリ以下にサンプリングレート別にディレクトリを用意します。
- データーベースを 16kHz で作成しても、同期のとれた（16kHz と時間情報が同じ）音声ファイルを、任意のサンプリングレートディレクトリの同じ（相対的に同じ）位置に配置することにより、合成音を任意のサンプリングレートで出力することができます。設定方法は、インストールマニュアルをご参照ください。
- 16kHz 以外のサンプリングレートのディレクトリには、音声ファイル以外のファイルを配置する必要はありません。
- seg ディレクトリ以下は、データベース作成 GUI を使用してデータベースを作成すると、自動的に作成されます。
- hts ディレクトリ以下は、HTS 学習 GUI を使用して HTS 学習を行うと、自動的に作成されます。
- settings ディレクトリは、XIMERA Version1.1 のリリースメディアからコピーしてご使用ください。
- settings ディレクトリ以下は、辞書に関する情報です。原則的に設定ファイル以外は変更しないでください。ユーザー辞書等はこのディレクトリ以下に作成されます。

3． 音声切り出しツール（JSPSEG）使用方法

音声切り出しツールは、音声ファイルを 1 文単位に切り出します。また、テキストファイルの修正に利用します。使用方法は以下のとおりです。

このツールは、Javaで記述されています。あらかじめ、J2SE™ v1.4.2_04 がインストールされたマシンでご利用ください。

3.1 入力ファイルの準備

切り出し対象となる音声ファイルと、テキストファイルが必要となります。
それぞれのフォーマットは以下のとおりです。

表 3.1 音声切り出しツール　入力ファイル

音声ファイル	RIFF ヘッダー付きの PCM 音声ファイル 16bit 量子化 mono もしくは 32bit 量子化 mono サンプリング周波数はいずれでも可能 ファイルの拡張子は、.wav
テキストファイル （読み上げ原稿）	テキストファイル（読み上げ原稿） 漢字コードは、SJIS、EUC、JIS のいずれか 1 文を 1 行に記述 ファイル名は、(音声ファイル名).txt

＜注意＞

- 音声ファイルとテキストファイルは同じディレクトリ内に置いてください。
- 任意ファイル名が同じ音声ファイルとテキストファイルが揃っているとデータファイルをロードして正常に作業画面が表示されます。
- 「データベース作成ツール」の入力となる音声ファイルの周波数は、16kHz または 48kHz、16bit 量子化 mono です。

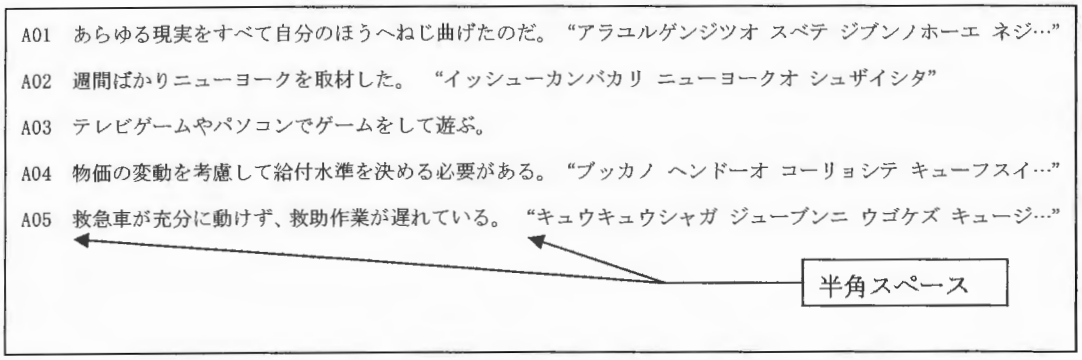
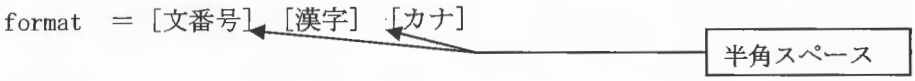


図 3.1 テキストファイル (DBname.txt) サンプル



・文中にスペースがある場合は” ” で囲んでください。
・漢字とカナは半角スペースで区切ってください。
・漢字のみで、カナが記述されていない場合は漢字からカナを内部的に作成します。
その場合、正しいカナが作成されない場合があります。

3．2 操作説明

3．2．1 起動方法

●Windows 版

Ximera110¥src¥dbtool¥jspseg¥segmentation-sentence.jar
をダブルクリックしてください。

●Linux 版

java -jar Ximera110/src/dbtool/jspseg/segmentation-sentence.jar
を実行してください。

次の画面が表示されます。

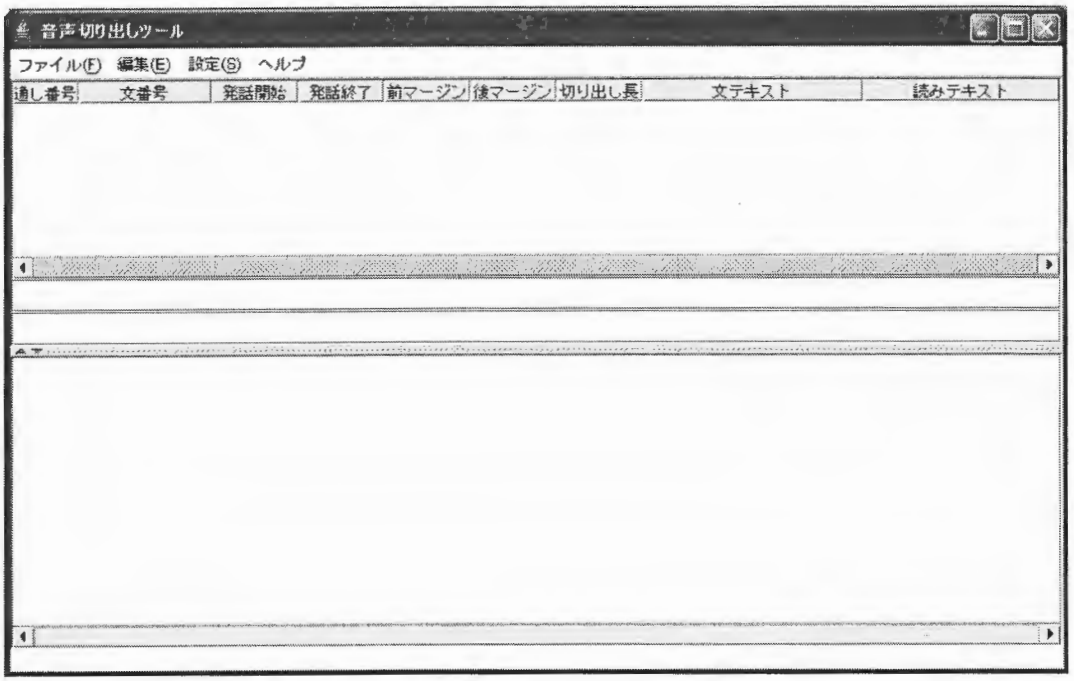


図 3．2 音声切り出しツールメイン画面（初期画面）

3．2．2 辞書のパス設定

切り出しツールを起動後、辞書を使用するかどうか選択して、辞書のパスを設定します。辞書を使用し辞書のパスを設定すると、ファイルロード時、カナが無い場合カナを作成し読みテキストに表示します。また、音声ファイルの切り出し（<3.2.11 切り出し>参照）を実行した時に、言語韻律情報（.pli）を出力することができます。

メニューバーより、【設定（S)】〔辞書使用〕にチェックマークが入っているのを確認します。

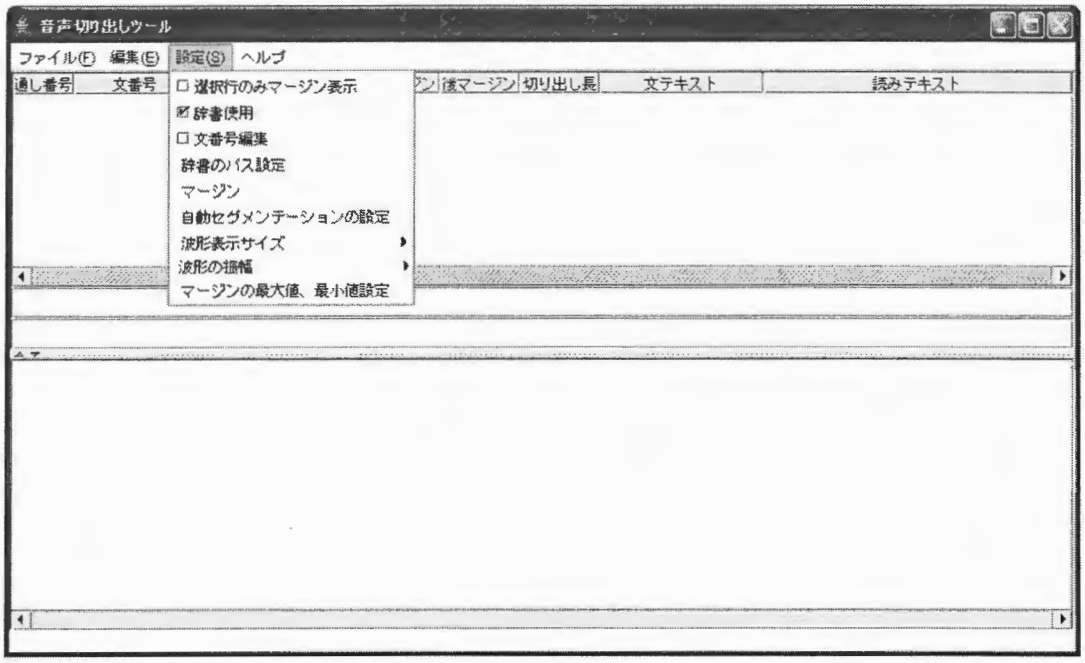


図 3．3 辞書使用設定画面

メニューバーより、【設定（S)】〔辞書のパス設定〕メニューを実行し、表示されたディレクトリ選択画面で xdata ディレクトリを選択します。

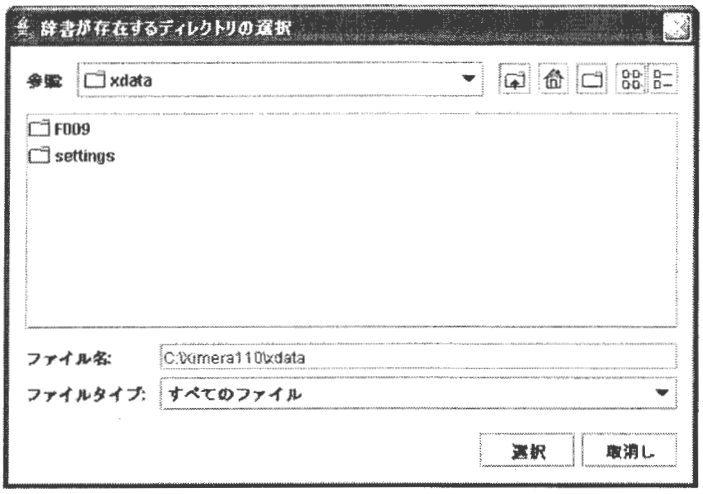


図 3．4 辞書のパス設定画面

＜注意＞　一度設定すれば記憶されるので、毎回設定する必要はありません。

３．２．３ ファイルのオープン

メニューバーより、【ファイル (F)】〔開く(Ctrl-O)〕で音声ファイル(.wav)を選択します。

ディレクトリ内に選択した音声ファイル(.wav)と同じ任意ファイル名のテキストファイル(.txt)が揃っていれば同時にオープンされ、それぞれ表示されます。

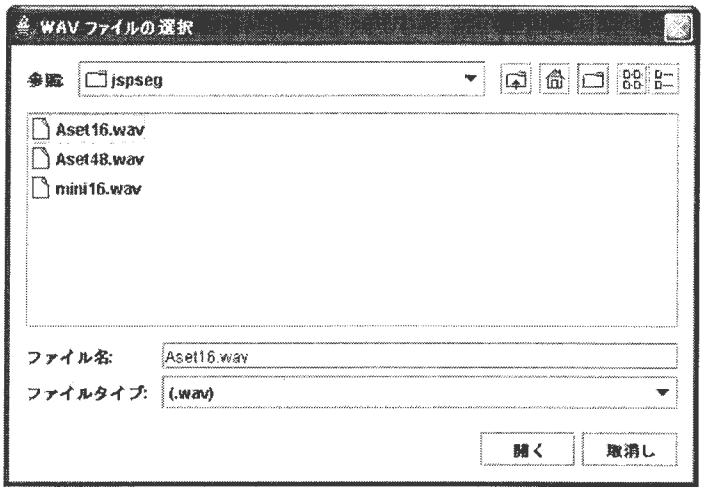


図 ３．５ 作業 wav ファイル選択画面

＜注意＞
選択した音声ファイル（.wav）と同じ任意ファイル名のセグメントファイル（.seg）が存在すればセグメント情報も表示されます。

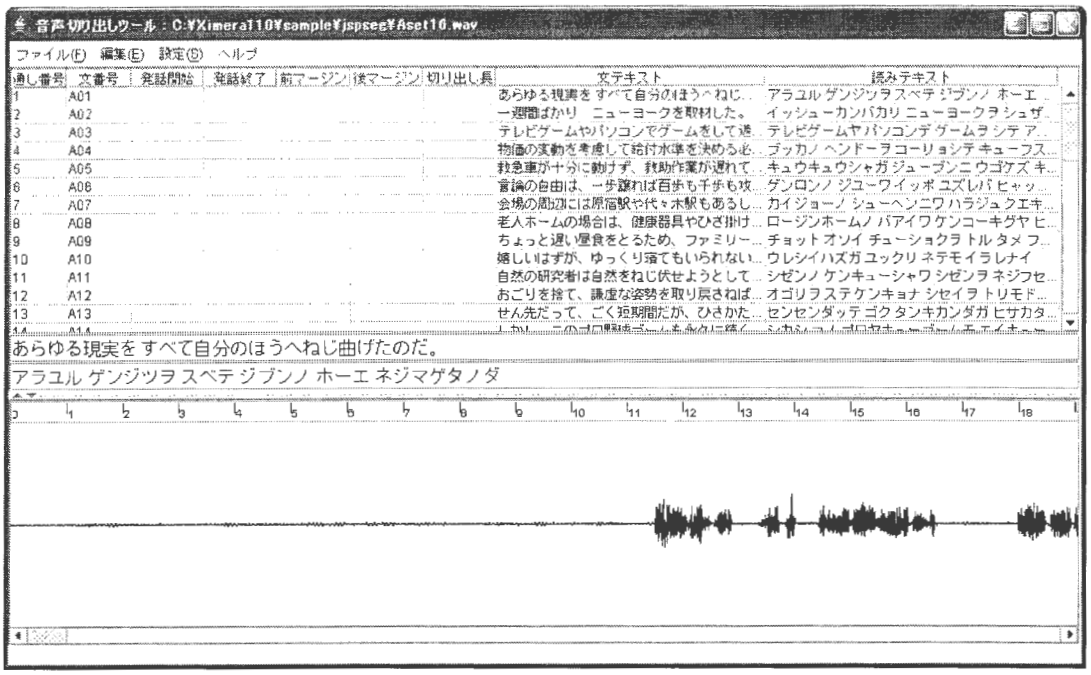


図 3．6 音声切り出しツール初期画面

図 3.6 の画面が表示されたら、<3.2.5 マージンの設定>を参照してください。

3．2．4 コマンドの機能と表示説明

以下の図3.7の項目ごとに機能と表示を説明します。

図 3．7 音声切り出しツール初期画面各項目名称

① タイトルバー

現在の作業ファイル名（音声ファイル名）が表示されます。

② メニューバー

表3．2 メニュー機能概要

	プルダウンメニュー	機能概要
ファイル(F)	開く (Ctrl-O)	データファイルを開きます。
	上書き保存 (Ctrl-S)	データファイルを上書き保存します。 未保存データは自動的にファイル名をつけて保存します。（<3.2.10 保存>を参照してください。）
	名前を付けて保存	データファイルに名前を付けて保存します。

	チェック	セグメントとテキストのチェックを行います。（<3.2.9 チェック>を参照してください。）
	終了（Ctrl-W）	音声切り出しツールを終了します。
編集（E）	元に戻す（Ctrl-Z）	直前に実行した操作を取り消して、元の状態に戻します。
	やり直し（Ctrl-Y）	〔元に戻す(Ctrl-Z)〕機能を使って取り消した操作を再度やり直します。
	自動セグメンテーションの実行	自動セグメンテーションを実行します。（<3.2.7 自動セグメンテーションの実行>を参照してください。）
	音声ファイルの切り出し	音声ファイルを切り出します。（<3.2.11 切り出し>を参照してください。）
	次のマージンの再生（Ctrl-P）	マージン部分の音声を再生して次のマージンへ移動します。
	文字列検索（Ctrl-F）	「文テキストのみ」「読みテキストのみ」「文テキストと読みテキストの両方」の3パターンで文字列検索をします。ステータスバーには検索結果件数が表示されます。
	マージンの検索（Ctrl-M）	任意に指定したマージン幅（ミリ秒単位）以上のマージンを検索します。
	次を検索（F3）	文字列検索・マージンの検索で検索した結果を、テキスト表示パネル上で通し番号順に選択し表示します。同期して波形も表示します。
	前を検索（Shift-F3）	文字列検索・マージンの検索で検索した結果を、テキスト表示パネル上で順に戻って選択し表示します。同期して波形も表示します。
設定（S）	選択行のみマージン表示	設定することにより、選択したセグメントのマージンのみ表示します。
	辞書使用	設定することにより、テキストファイルをロードした時、カナの記述がなければ漢字からカナを作成します。また設定することにより、音声を切り出した時、言語韻律情報ファイルを作成します。
	文番号編集	設定することにより、文番号を編集できます。

	辞書のパス設定	辞書のパスを設定します。（<3.2.2 辞書のパス設定>を参照してください。）
	マージン	セグメント分割時のマージン幅を、ミリ秒単位で設定します。（<3.2.5 マージンの設定>を参照してください。）
	自動セグメンテーションの設定	自動セグメンテーションに関する設定値を設定します。（<3.2.6 自動セグメンテーションの設定>を参照してください。）
	波形表示サイズ	
	波形表示サイズ設定	音声波形の表示サイズを任意に変更できます。<デフォルト値：50 ピクセル/秒>
	200 ピクセル/秒で表示 (Ctrl-F7)	音声波形を 200 ピクセル/秒で表示します。
	50 ピクセル/秒で表示 (Ctrl-F8)	音声波形を 50 ピクセル/秒で表示します。
	波形の振幅	
	波形の振幅倍率設定	音声波形の振幅倍率を任意に変更できます。<デフォルト値：1.0>
	振幅を 2 倍にする	音声波形の振幅を現在の 2 倍に変更します。
	振幅を 1/2 倍にする	音声波形の振幅を現在の 1/2 倍に変更します。
	マージンの最大値、最小値の設定	マージンの長さ（幅）の設定範囲を指定します。<3.2.5 マージンの設定>を参照してください)
ヘルプ	バージョン情報	バージョン情報を表示します。

③ テキスト表示パネル

表 3.3 テキスト表示パネル機能概要

通し番号	ロードしたテキストファイルに記述されている文をカウントし、1 から順に連番をつけて表示します。
文番号	ロードしたテキストファイルに記述されている文の番号を表示します。

発話開始（単位秒）	発話開始時刻を小数点以下3桁で表示します。
発話終了（単位秒）	発話終了時刻を小数点以下3桁で表示します。
前マージン（単位秒）	前マージンの長さ（幅）を小数点以下3桁で表示します。
後マージン（単位秒）	後マージンの長さ（幅）を小数点以下3桁で表示します。
切り出し長（単位秒）	切り出し開始時刻（＝発話開始－前マージン）から切り出し終了時刻（＝発話終了＋後マージン）までの長さを小数点以下3桁で表示します。
文テキスト	ロードしたテキストファイルに記述されている漢字かな混じり文を表示します。
読みテキスト	ロードしたテキストファイルに記述されているカナを表示します。記述がない時は辞書使用を設定した場合のみ漢字かな混じり文よりカナを作成して表示します。

詳細は、<3.2.8 セグメンテーションとテキストの編集>を参照してください。

④ テキスト修正パネル

読み間違い、原稿記述ミスなどで、発話内容と文が一致していない場合、テキストを直接編集します。該当テキストにカーソルを合わせマウス左ボタンをクリックすると編集可能になります。テキスト修正パネルでの修正は、テキスト表示パネルに同期して反映されます。

詳細は、<3.2.8 セグメンテーションとテキストの編集>を参照してください。

⑤ 波形表示パネル

音声波形を表示します。

また、自動セグメンテーションの実行によりセグメントを分割し、開始位置を赤、終了位置を青のマージン（帯）で表示します。

範囲指定された波形や、波形の前後にあるマージンをマウス左ボタンでダブルクリックすると音声が再生されます。もう一度聞く場合はマウス右ボタンによりポップアップメニューを表示し再生します。

詳細は、<3.2.8 セグメンテーションとテキストの編集>を参照してください。

⑥ ステータスバー

ステータスを表示します。

３．２．５ マージンの設定

＜3.2.3 ファイルのオープン＞で初期画面が表示されたら、自動セグメンテーションを実行する前に、【設定（S)】〔マージン〕でマージンの設定を行います。

デフォルト値は 1000 ミリ秒に設定されています。

図 ３．８ マージン幅の設定画面

＜注意＞

マージン幅の一括変更ではなくて設定です。設定後に【編集】〔自動セグメンテーションの実行〕、または自動セグメンテーションを実行後マウス右ボタンでポップアップメニューを表示し〔セグメントを分割〕の動作を実行したときに有効になります。

3．2．6 自動セグメンテーションの設定

＜3.2.5 マージンの設定＞でマージン設定後、自動セグメンテーションを実行する前に、【設定(S)】〔自動セグメンテーションの設定〕で自動セグメンテーションの設定を行います。

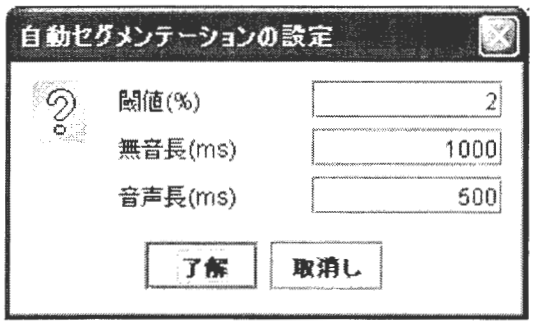


図 3．9 自動セグメンテーションの設定画面 <デフォルト値>

自動セグメンテーションの設定値は以下の表を参考にしてください。

表 3．4 自動セグメンテーションの設定内容

閾値 （%）	指定された%以下をノイズ（無音）と判断します。 音声ファイルのノイズの大きさに合わせて、値を調整します。
無音長 （ミリ秒）	指定ミリ秒以上の無音があれば、セグメントを分割します。 1 文の発話間隔を指定します。
音声長 （ミリ秒）	指定ミリ秒以上の有声音があれば、セグメントを作成します。 1 文の発話時間を指定します。

３．２．７ 自動セグメンテーションの実行

【編集（E)】〔自動セグメンテーション〕を実行すると、事前の設定に従ってセグメントを分割し、分割した結果を画面に表示します。

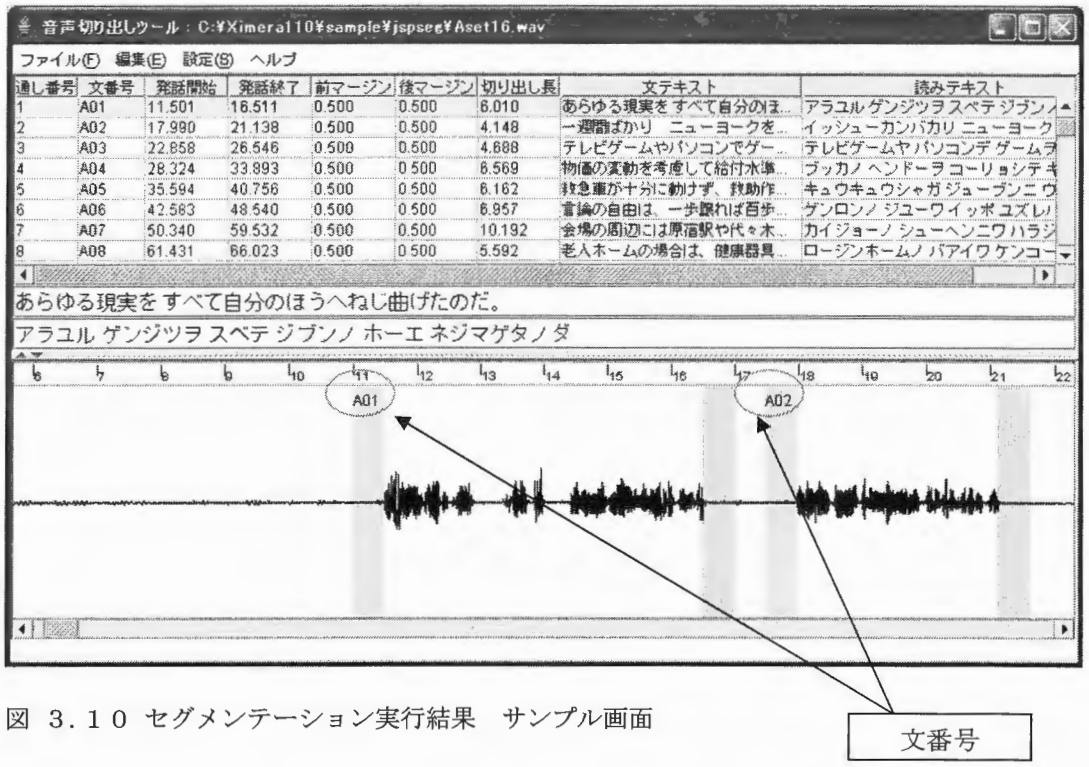


図 ３．１０ セグメンテーション実行結果　サンプル画面

マーカー表示された自動セグメンテーションの結果を参照し、必要に応じて設定値を調整後、再度【編集】〔自動セグメンテーションの実行〕を行ってください。

表 ３．５ 自動セグメンテーションの設定値変更例

マージンの幅を変更したい場合	【設定（S）】〔マージン〕よりマージンの設定を変更します。
ノイズを音声と判断している場合	【設定（S）】〔自動セグメンテーションの設定〕より、閾値の設定を大きくします。
文途中のポーズでセグメント分割している場合	【設定（S）】〔自動セグメンテーションの設定〕より、無音長を大きくします。
咳払いなどの短い音声をセグメントとして分割している場合	【設定（S）】〔自動セグメンテーションの設定〕より、音声長を大きくします。

３．２．８ セグメンテーションとテキストの編集

自動セグメンテーション実行後、各パネル（図 3.7 音声切り出しツール初期画面参照）からセグメントとテキストの編集を行います。

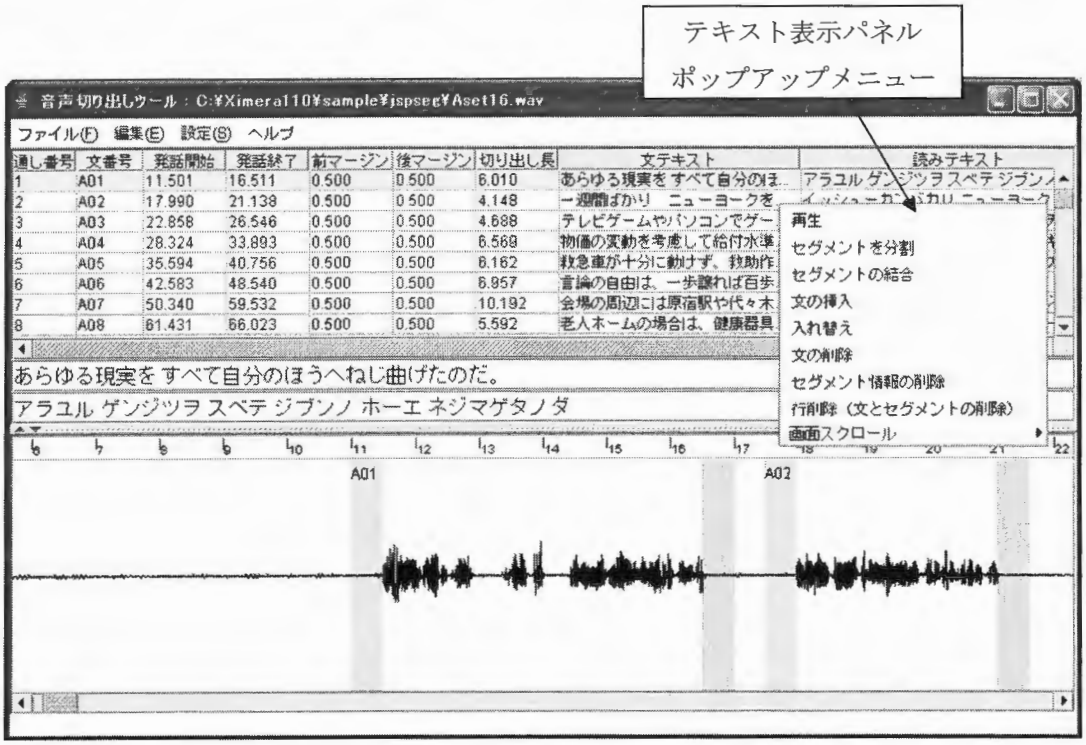


図 ３．１１ 自動セグメンテーションとテキストの編集　サンプル画面

(1) 波形表示

テキスト表示パネル上で任意の１行を選択すると、該当セグメントが波形表示パネルの中心に示されます。

(2) テキストの編集

読み間違い、原稿記述ミスなどで、発話内容と文が一致していない場合、テキストを直接編集します。<F12>キーでテキスト修正パネルに移動するか、テキスト修正パネルの該当テキストにカーソルを合わせマウス左ボタンでクリックすると編集可能になります。

編集したテキストはテキスト表示パネルの文テキスト・読みテキストに反映されます。

(3) セグメンテーション位置の編集

・波形表示パネルのマージン（カラーの帯）にデフォルトカーソルを合わせてマウス

右ボタンのドラッグで左右に移動することができます。テキスト表示パネルの発話開始時刻と切り出し長、または発話終了時刻と切り出し長が自動的に変更されます。

・個別にマージン幅を任意の幅に変更することができます。マージン（カラーの帯）の端（境界線）にカーソルを合わせると表示がデフォルトカーソルからハンドカーソルに変わります。この時点でマウス右ボタンでドラッグするとマージン幅を変更することができます。

ただし、設定できる長さ（幅）は【設定（S)】〔マージンの最大値、最小値設定〕で指定した範囲内に限ります。
デフォルト値は、最小値 300 ミリ秒、最大値 1000 ミリ秒に設定されています。

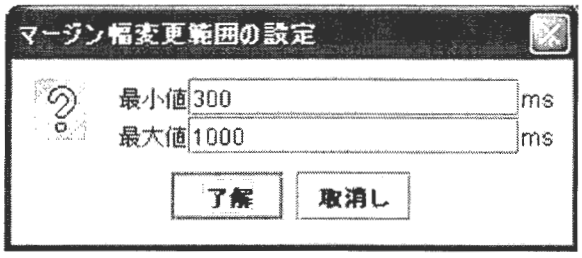


図 3.12 マージン幅変更範囲の設定画面

＜注意＞
マージン幅を個別に変更したあと、再度【編集】〔自動セグメンテーションの実行〕を行うと、図 3.8 マージン幅の設定画面で設定したマージン幅が優先され、個別変更が取り消されます。

(4) 音声の再生

① 波形表示パネル上で再生したい任意の範囲をマウス左ボタンでドラッグすると、指定範囲が水色に変わります。マウス左ボタンを離すと同時に音声が再生されます。もう一度聞く場合は波形表示パネル上でマウス右ボタンによりポップアップメニューを表示し再生します。

② 波形表示パネル上で、赤・青の順のマージンで挟まれた波形をマウス左ボタンでダブルクリックすると指定範囲が水色に変わり音声が再生されます。もう一度聞く場合はマウス右ボタンによりポップアップメニューを表示し再生します。
同様にマージンをマウス左ボタンでダブルクリックしてもマージンの範囲のみを再生します。

③ テキスト表示パネルで任意の 1 行を選択後、マウス右ボタンによりポップアップメニューを表示し、再生することも可能です。この場合、前後のマージンを含めた範

囲が再生されます。
複数行を選択した場合は一番上の行のみが再生されます。

(5) セグメントを分割

テキスト表示パネルで1行を選択し、マウス右ボタンでポップアップメニューを表示し〔セグメント分割〕を実行すると、分割数（2～4）に応じて、セグメントを等分に分割します。文数はそのままで、セグメント数が増加します。

(6) セグメントの結合

テキスト表示パネルで連続した2行を選択し、マウス右ボタンでポップアップメニューを表示し〔セグメントの結合〕を実行すると、1行目の開始から2行目の終了までを1つのセグメントにします。セグメント数が1つ減少します。

(7) 文の挿入

マウス右ボタンでポップアップメニューを表示し〔文の挿入〕を実行すると、選択した行の上に空文を1行挿入します。文数が1つ増加します。

(8) 入れ替え

任意の2行を選択し（連続した2行でない場合は、Ctrl キーで選択）、マウス右ボタンでポップアップメニューを表示し〔入れ替え〕を実行すると、テキストが入れ替わります。

(9) 文の削除

マウス右ボタンでポップアップメニューを表示し〔文の削除〕を実行すると、選択された（複数）行のテキストを削除します。セグメント数は変わりません。

(10) セグメント情報の削除

マウス右ボタンでポップアップメニューを表示し〔セグメント情報の削除〕を実行すると、選択された（複数）行のセグメント情報を削除します。文数は変わりません。

(11) 行削除（文とセグメントの削除）

マウス右ボタンでポップアップメニューを表示し〔行削除（文とセグメントの削除)〕を実行すると、選択された（複数）行の文とセグメント情報を同時に削除します。

(12) 画面スクロール

マウス右ボタンでポップアップメニューを表示し〔画面スクロール〕を実行すると、マージンの表示位置を移動できます。

・前マージンを左に表示　　<F11>キー
・前マージンを中央に表示　　<F1>キー
・後マージンを中央に表示　　<F2>キー
・後マージンを右に表示　　<Shift-F11>キー

３．２．９ チェック

セグメンテーションとテキストの編集後、正しく編集されているかチェックを行います。【ファイル（F)】〔チェック〕を実行します。

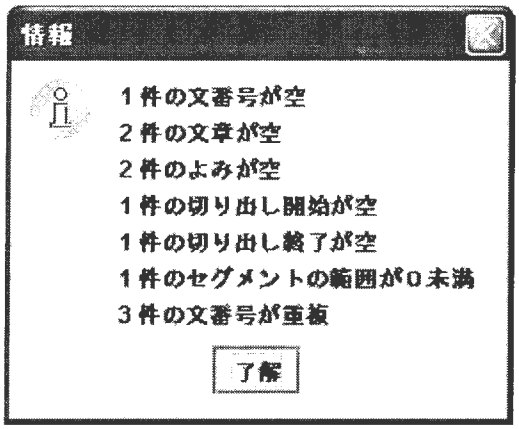


図 ３．１３ チェック結果画面サンプル

セグメント数とテキスト数が一致しているかをチェックします。

また、各セグメントにおいて、“切り出し開始時刻 ＜ 切り出し終了時刻” になっているかをチェックします。

エラーがない場合は、ステータスバーに、

「チェック完了 エラー無し」というメッセージが表示されます。

エラーがある場合は、了解ボタンを押下するとチェック結果画面に記載された先頭の情報（図 3.13 の例では“１件の文番号が空”）へテキスト表示パネル上で移動します。

この時点でエラーを修正し、再度チェックを実行し情報に従って次のエラー修正を行います。

３．２．１０ 保存

保存を実行すると、下記ファイルが保存されます。

表 ３．６ 切り出しツール　保存ファイル

テキストファイル	文テキストの情報を１ファイルに出力 １文１行に出力 拡張子は、.txt
セグメントファイル	セグメント情報を１ファイルに出力 拡張子は、.seg

＜注意＞

セグメント情報とは、切り出し開始、発話開始、発話終了、切り出し終了時刻の４つの情報です。各時刻の単位は、＜秒＞です。

```
11.001 11.501 16.511 17.011
17.49  17.99  21.138 21.638
22.358 22.858 26.546 27.046
27.824 28.324 33.893 34.393
35.094 35.594 40.756 41.256
42.083 42.583 48.54  49.04
49.84  50.34  59.532 60.032
60.931 61.431 66.023 66.523
67.49  67.99  73.16  73.66
74.185 74.685 79.839 80.339
81.069 81.569 84.962 85.462
86.336 86.836 91.73  92.23
・・・・・
```

図 ３．１４ セグメントファイル（.seg）サンプル

(1) 上書き保存

【ファイル（F)】〔上書き保存〕を実行すると、ロードしている音声ファイル名の拡張子をそれぞれ上記に変更し、保存します。＜図 3.12 ＞のファイルを例にすると、“Aset16.txt”と“Aset16.seg”となります。

(2) 名前を付けて保存

【ファイル (F)】〔名前を付けて保存〕を実行すると、任意のファイル名に上記の拡張子を付加し、保存します。

３．２．１１ 切り出し

音声ファイルを切り出し、ディスクに保存します。
【編集 (E)】〔音声ファイルの切り出し〕を選択後ファイル名の先頭に付加する文字列を入力、ファイル名の後尾に付加する文字列を入力して出力先ディレクトリを指定します。

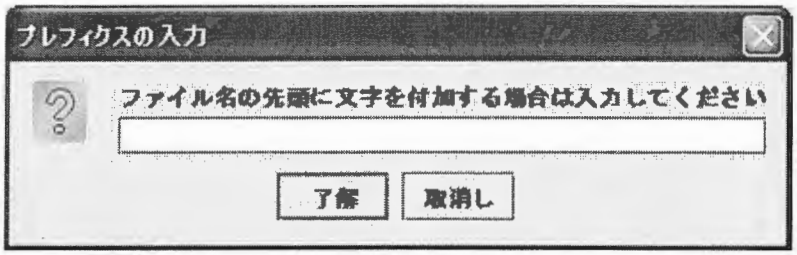


図 ３．１５ 切り出しファイル名　先頭文字列指定画面

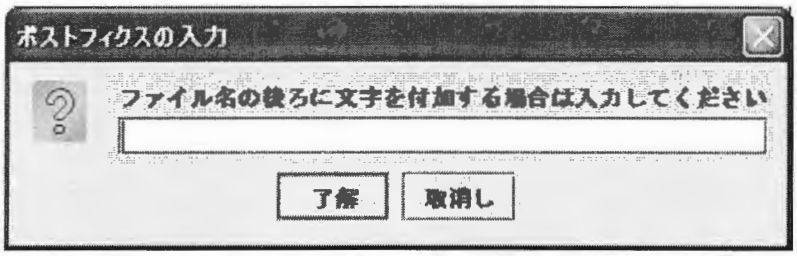


図 ３．１６ 切り出しファイル名　後尾文字列指定画面

上記画面で付加する文字を入力後、了解ボタンを押下しその後、出力先ディレクトリを選択すると、以下の仕様で切り出しを実行します。
※ファイル名に文字を付加する必要のない場合は、了解のみを押下します。

- １セグメント単位に音声ファイルを切り出します。
- ファイル名は、文番号に上記で入力した文字を付加したファイル名となります。
- 切り出された音声ファイルのフォーマットは、入力ファイルと同じになります。
- テキスト情報があれば音声ファイルと共に言語韻律情報ファイルとカナファイルをそれぞれ作成し出力します。
- 作成し出力されたカナファイルの文字コードは OS に依存します。(Windows であれば SJIS、Linux であれば EUC で出力されます。)

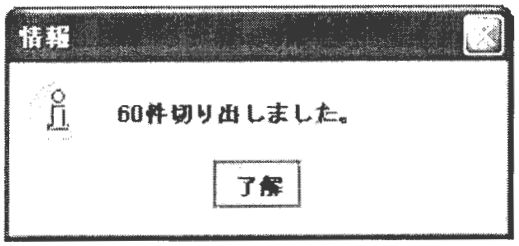


図 3.17 切り出し処理終了画面

表 3.7 切り出し・作成出力ファイル

ファイル	ファイル名	出力
切り出した音声ファイル	指定先頭文字列（文番号）指定後尾文字列.wav	セグメントに従って出力
言語韻律情報ファイル	指定先頭文字列（文番号）指定後尾文字列.pli	漢字より作成した言語韻律情報
カナファイル	指定先頭文字列（文番号）指定後尾文字列.kna	読みを出力

＜注意＞

辞書のパスが正しく設定されていなければ言語韻律情報（.pli）は出力されません。（＜3.2.2 辞書のパス設定＞を参照してください。）

4． サンプリングレート変換ツール（srconv）使用方法

収録した音声とデータベース作成に使用する音声ファイルのサンプリング周波数が異なる場合、サンプリングレート変換ツール srconv を使用し、変換します。

4．1 実行体のパス

Windows の場合
Ximera110¥bin¥windows¥srconv.exe

Linux の場合
Ximera110/bin/linux/srconv

4．2 実行方法

実行方法、オプションの使い方は以下のとおりです。

```
srconv［オプション］［入力ファイル　出力ファイル］
   オプション []内はデフォルト値
    -f     INT .....出力のサンプリング周波数[16000]
    -of    STR .... 出力のファイルサンプル形式(ulaw|alaw|short|float)[short]
    -a     FLOAT ...倍率[1.000]
    -l     FILE ....ファイルリスト[]
    -di    DIR .... 入力ディレクトリ[.]
    -do    DIR .... 出力ディレクトリ[.]
    -x     STR .... 入力ファイル拡張子[wav]
    -tr    INT .... トレースレベル[0]
    -h          .... 使用方法を表示
```

入力ファイルは RIFF ヘッダー付の PCM 音声ファイルを指定してください。
サンプリング周波数変換は、48kHz から 16kHz への変換にご使用ください。それ以外の変換はサポート外です。

使用例1（1ファイルを変換）

```
> srconv  -f  16000  in.wav  out.wav
```

上記例では、in.wav ファイルを、サンプリングレート 16kHz にサンプリング周波数変換し、out.wav に出力します。

使用例2（複数ファイルを変換）
変換対象となる音声ファイルリストを作成します。ファイル名は任意です。

ファイルリスト flist.txt サンプル

```
A01
A02
A03
...
```

ファイルリストを指定して一括変換します。

```
> srconv  -f 16000  -l  flist.txt  -di wav48  -do wav16
```

上記例では、wav48 ディレクトリ以下の A01.wav、A02.wav、A03.wav… を 16kHz にサンプリング周波数変換し、wav16 ディレクトリ以下に A01.wav、A02.wav、A03.wav…を作成します。

5． データベース作成手順

5．1 作成についての制限事項

Windows、Linux 共に、1 ファイルのサイズが約 2GB 以下でないといけないという制限があります。

データベース作成時の中間ファイルが 2GB を超えないようにするため、以下の制限があります。

作成するデータベースの音声コーパスは 30 時間以下

16kHz の音声ファイルの場合、音声ファイルのディスク使用量が約 3.45GB 以下を目安としてください。

音声ファイルのサンプリングレートがいずれであっても、音声ファイルの合計時間が 30 時間以下になるようにしてください。

30 時間以上のデータベースを作成するには、分割したデータベースを作成し、マージを行ってください。マージの方法については、「5.5 データベースのマージ方法」をご参照ください。

５．２ 入力ファイル

データベース作成に必要なファイルは以下のとおりです。

表 ５．１ データベース入力ファイル

音声ファイル	1 文単位に分割した音声ファイル RIFF ヘッダー付 サンプリングレートは 16kHz または 48kHz ファイルの拡張子は、wav
カナファイル	音声ファイルの発話内容をカタカナで記述 ファイル名は、(音声ファイル名).kna 漢字コードは、SJIS、EUC、JIS のいずれか （アクセント記号「'」は存在してもよい）

5．3 入力ファイルをディレクトリに配置

音声切り出しツールを使用して作成した音声ファイル(*.wav)とカナファイル(*.kna) を以下のディレクトリにおきます。(「2. ディレクトリ構成」参照)

xdata¥〔話者名〕¥corpus¥〔サンプリングレート名〕¥〔サブコーパス名〕¥

〔サブコーパス名〕以下は、管理を容易にするため、複数のディレクトリに分割してもかまいません。また、何階層でもかまいません。

＜例＞

〔サブコーパス名〕¥A¥A_01.wav
〔サブコーパス名〕¥A¥A_01.kna
〔サブコーパス名〕¥B¥B_01.wav
〔サブコーパス名〕¥B¥B_01.kna
〔サブコーパス名〕¥C¥01¥C_01.wav
〔サブコーパス名〕¥C¥01¥C_01.kna
〔サブコーパス名〕¥C¥02¥C_02.wav
〔サブコーパス名〕¥C¥02¥C_02.kna

＜注意＞

データベースは、音声サブコーパスを選択（複数可）して作成します。データベース作成時に選択する最小単位を音声サブコーパス単位としてください。

５．４ データベース作成 GUI

５．４．１ データベース作成 GUI の起動

Windows の場合

Ximera110¥src¥dbtool¥dbmake-gui¥AutoDBmake.bat

Linux の場合

Ximera110/src/dbtool/dbmake-gui/AutoDBmake.sh

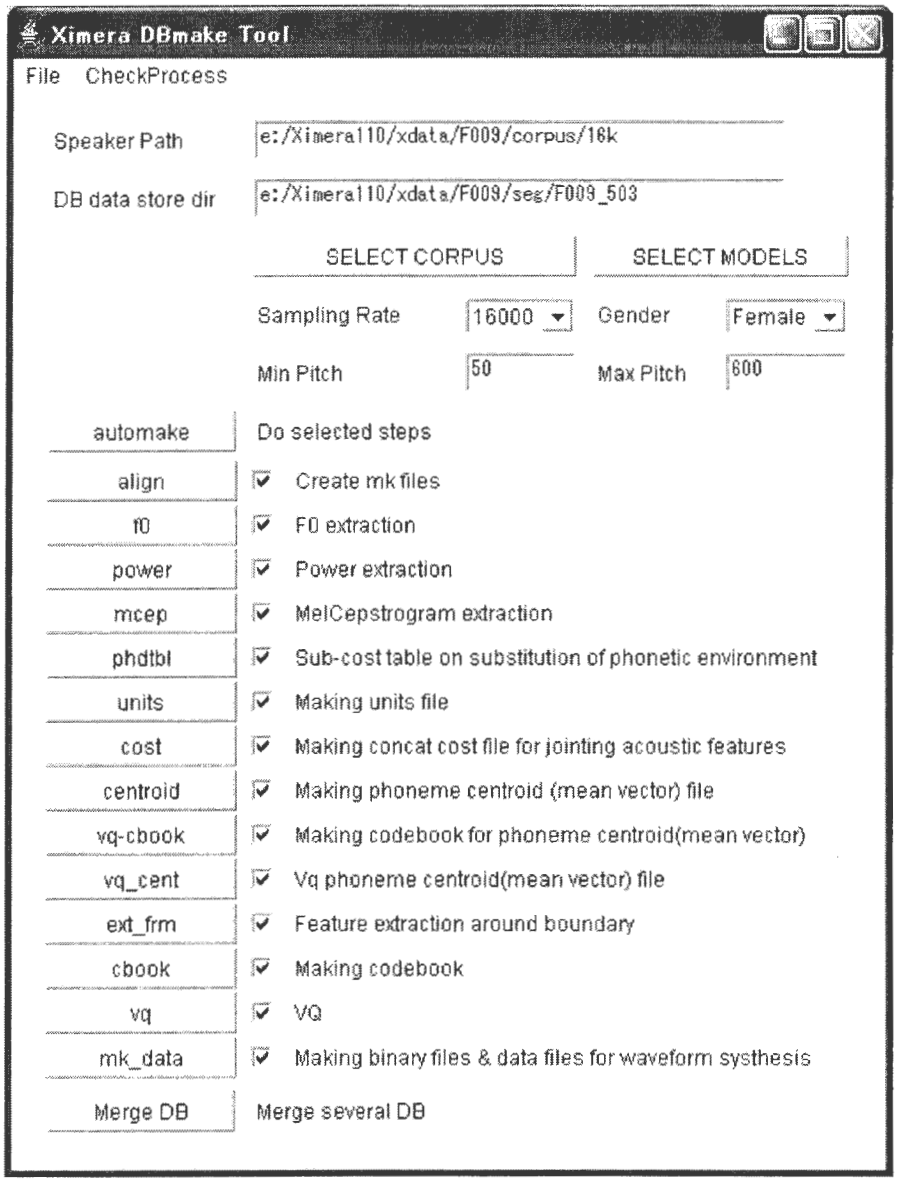


図 ５．１ データベース作成 GUI 初期画面サンプル

5．4．2 データベース作成 GUI の設定

(1) 入力エリアに下記を入力してください。

Speaker Path　　選択するサブコーパスが存在するディレクトリを入力
DB data store dir　作成するデータベースパスを入力

(2) SELECT CORPUS

SELECT CORPUS ボタンを押下すると、[Speaker Path] ディレクトリ直下のディレクトリが表示されます。
データベースに含めたいサブコーパスディレクトリを選択してください。(複数選択可)

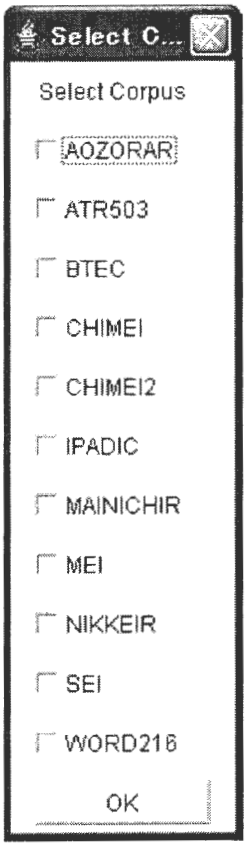


図 5．2 コーパス選択画面サンプル

(3) SELECT MODELS

align の工程で使用するモデルを選択します。SELECT MODELS ボタンを押下します。
デフォルトで表示されているモデルをご使用ください。

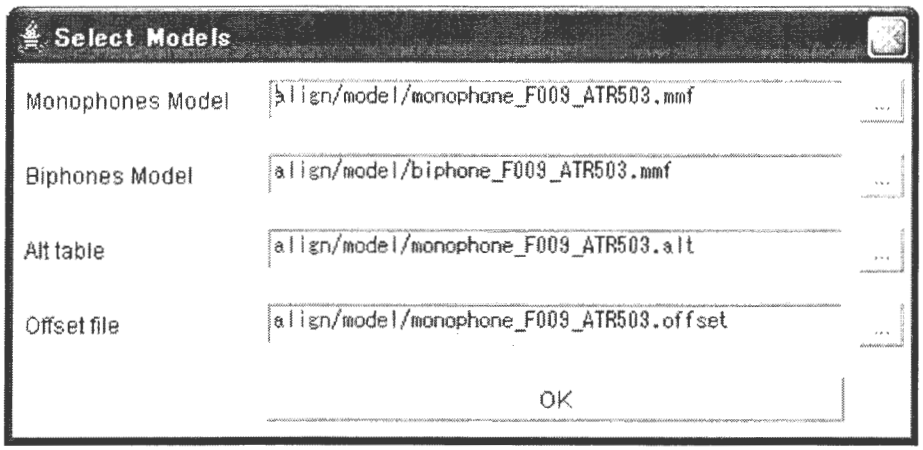


図 5.3 モデル設定画面サンプル

モデル設定画面では、以下のファイルを選択します。

表 5.2 モデル設定項目内容

Monophones Model	monophones モデルのパスを入力
Biphones Model	biphones モデルのパスを入力
Alt table	代替音素に関する定義ファイルのパスを入力
Offset file	オフセットファイルのパスを入力 ファイルの指定がなければ、align の工程でオフセットが 0 となります。

(4) サンプリングレート、F0 範囲の設定

以下の表の項目を設定します。

表 5.3 その他設定項目

Sampling Rate	入力する音声ファイルのサンプリングレートを選択します。
Gender	Female、Male のいずれかを選択します。
Min Pitch	F0 抽出時の最低値。f0 の工程のパラメータとなります。
Max Pitch	F0 抽出時の最大値。f0 の工程のパラメータとなります。

(5) 工程の選択

データベース作成GUIより選択実行できる工程は､機能的に大きく2つに分かれます。

align ～ mcep までの工程は、サブコーパス単位で実行する工程です。

phdtbl ～ mk_data までの工程は、データベース単位で実行する工程です。

(5-1)align ～ mcep

この4つの工程は、音声ファイルと同じディレクトリにそれぞれ下記拡張子のファイルを作成します。

表 5.4 各工程出力ファイル

工程名	作成するファイル
align	ラベルファイル(*.xmk)
f0	F0ファイル(*.xf0)
power	パワーファイル(*.pow)
mcep	ケプストラムファイル(*.mcep)

これらの工程はそのサブコーパスに対して一度実行すると、再度実行する必要はありません。ただし、音声ファイル(*.wav)やカナファイル(*.kna)が変更になった場合は、再度実行してください。

(5-2)phdtbl ～ mk_data

作成したいデータベースの単位で、phdtbl ～ mk_data まですべての工程を実行してください。[DB data store dir] ディレクトリに必要なファイルが作成されます。SELECT CORPUS で選択した音声ファイルすべてに対して align ～ mcep の工程が実行されてない場合は、先にそのサブコーパスに対して上記の工程を実行してください。

5．4．3 データベース作成の実行

automake ボタンの押下で、選択した工程を実行します。
すべての工程が完了すれば、データベース作成完了です。

※ およその実行時間は次のとおりです。
使用マシン： CPU Pentium(R) 4 CPU 2.40GHz
メモリ 2GB
作成データベース：文章 503 文（16kHz の音声ファイル約 35 分）
実行時間：”align”から”mk_data”までの工程で約 1 時間 35 分

5．4．4 F0 ファイル、ラベルファイルを修正する場合

F0、ラベルファイルを修正する場合は、align、f0 の工程を実行した後、F0・ラベル修正ツールで修正し、power 以降の工程を実行してください。
また、既にデータベース作成が完了している場合は、F0・ラベル修正を行った後、units 以下の工程を実行してください。

5．5 データベースのマージ方法

作成済みデータベースをマージする場合、以下の手順で行います。
Windows、Linux 共通です。Linux コマンドで説明を記述しています。Windows の場合は、コマンドを読み替えて実行してください。

(ア)　データベースディレクトリの作成

作成するデータベース名のディレクトリを作成します。

```
[seg]$ mkdir MERGEDB
```

※　MARGEDB に作成するデータベース名を入力してください。

(イ)　マージするデータベースの指定

マージするデータベースを以下の 2 つのファイルに記述します。

- [MARGEDB]/mcep24+f0_ccs.list
- [MARGEDB]/mcep24+f0_clen.list

mcep24+f0_ccs.list ファイル編集例

```
/home/ximera/Ximera110/xdata/F009/seg/ATR503/mcep24+f0.ccs
/home/ximera/Ximera110/xdata/F009/seg/BTEC/mcep24+f0.ccs
/home/ximera/Ximera110/xdata/F009/seg/IPADIC/mcep24+f0.ccs
```

＜注意＞

- マージするデータベースは、既に作成済みである必要があります。
- マージするデータベースをすべて記述してください。
- 上記例では、「ATR503」「BTEC」「IPADIC」データベースをマージします。

Mcep24+f0_clen.list ファイル編集例

```
/home/ximera/Ximera110/xdata/F009/seg/ATR503/mcep24+f0.clen
/home/ximera/Ximera110/xdata/F009/seg/BTEC/mcep24+f0.clen
/home/ximera/Ximera110/xdata/F009/seg/IPADIC/mcep24+f0.clen
```

＜注意＞

- mcep24+f0_ccs.list ファイルに記述したデータベースの順番と同じ順番で記述してください。

（ウ） 実行

データベース作成 GUI 画面を開きます。

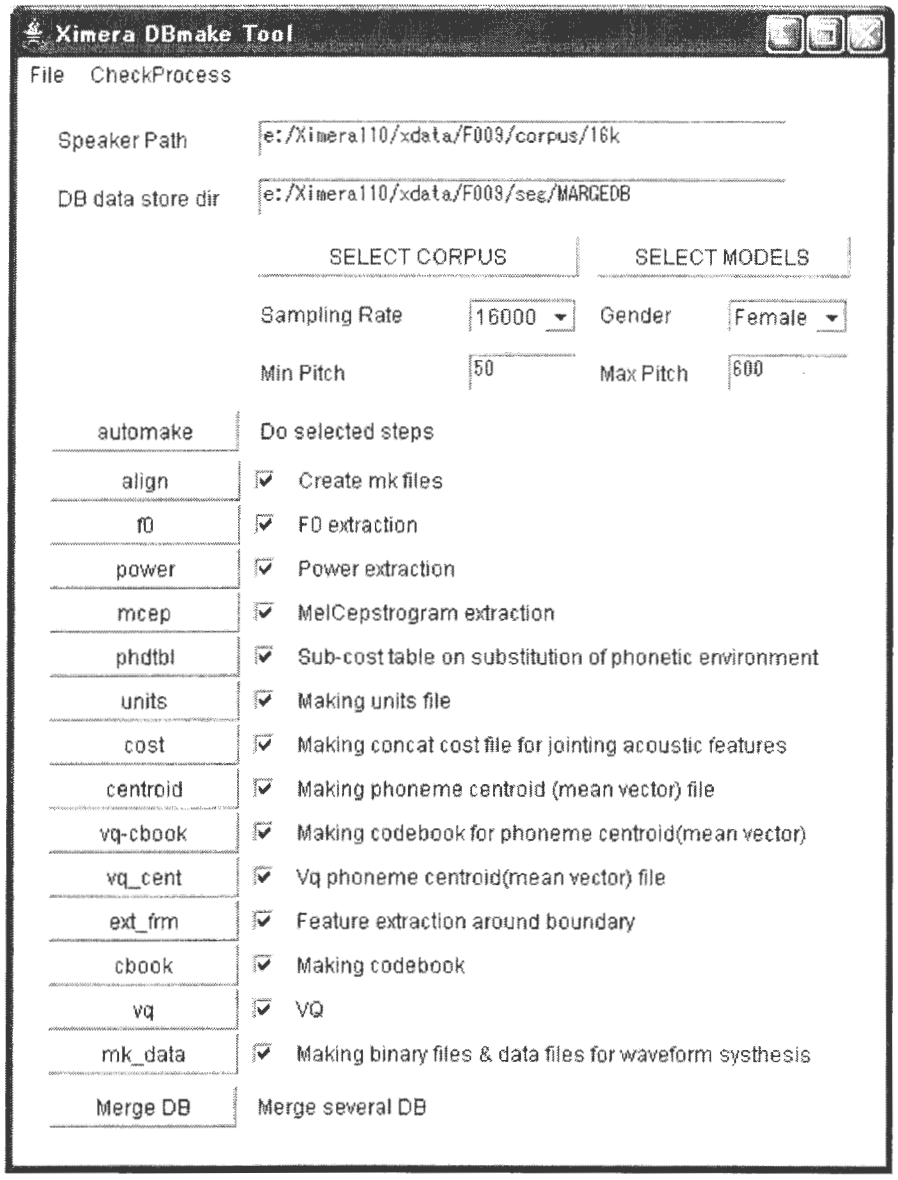


図 5．4 データベースマージ初期画面

DB data store dir に、手動で作成したディレクトリを入力し、
Merge DB ボタンを押下します。
（align ～ mk_data までのチェックボックスは無視されます。）
マージ実行中は、実行している工程が着色表示されます。

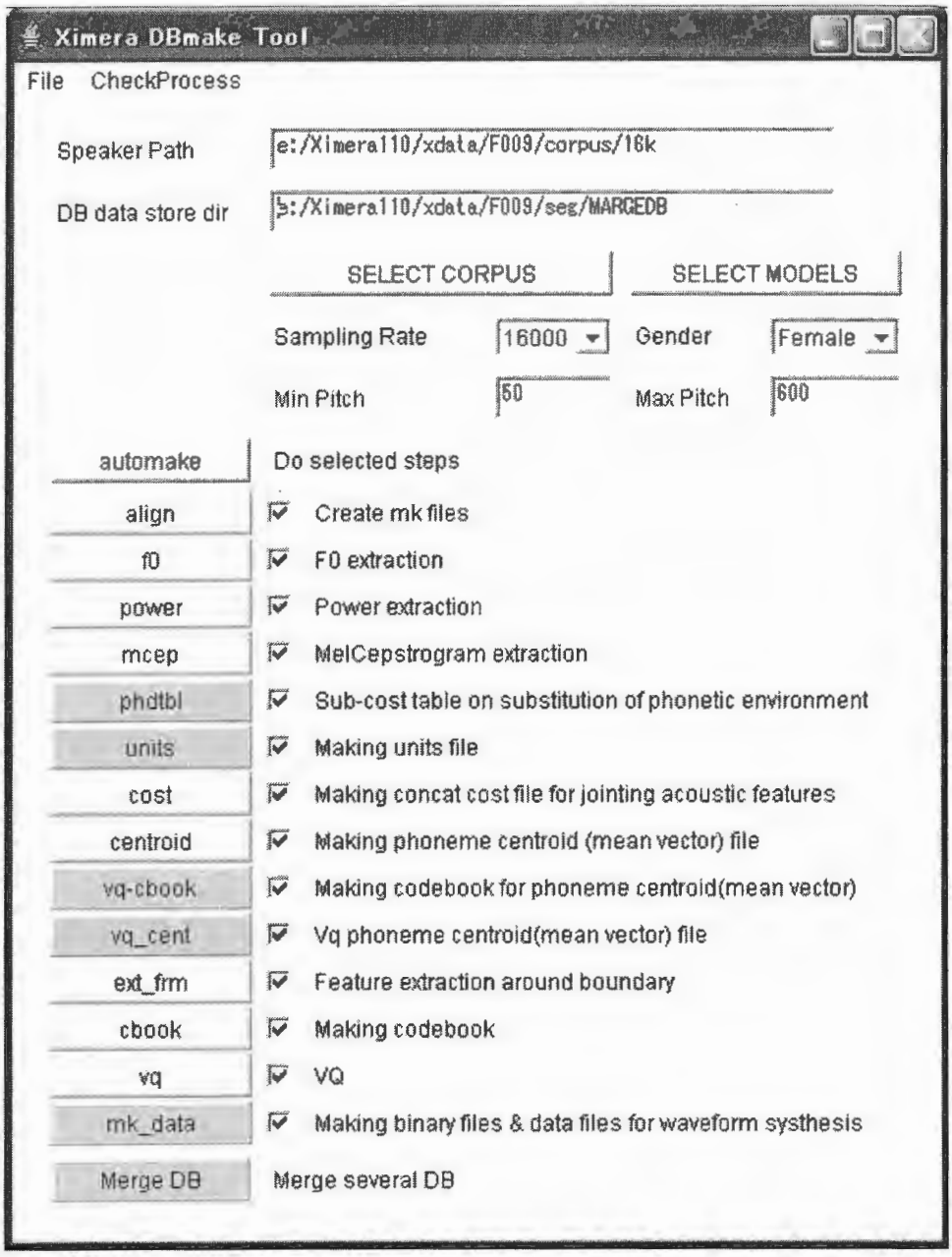


図 5.5 データベースマージ完了画面サンプル

※ およその実行時間は次のとおりです。

使用マシン： CPU Pentium(R) 4 CPU 2.40GHz
メモリ 2GB

マージデータベースサイズ：約 47.6 時間

実行時間： 約 3 時間

(エ)　ファイルコピー

cbdist.mat のコピー

データベースマージでは、vq の工程を省略しているため、既に作成済みのデータベースディレクトリより、cbdist.mat をコピーします。

```
[seg]$ cp ATR503/cbdist.mat  MARGEDB/cbdist.mat
```

※　上記は、既存のデータベース ATR503 より、マージにより作成したデータベース MARGEDB にコピーする場合の例です。

※　同じ話者のデータベースからコピーを行ってください。

5.6 ファイル編集

XIMERA エンジンでコーパスにアクセスするために、data_wvc.txt ファイルを編集します。ファイルのパスは以下のとおりです。

seg¥[データベース名]¥data_wvc.txt

```
-f 16000
-shift 5.0
-f0shift 10.0
-wavdir ../../corpus/16k
-wavsign wav
```

＜注意＞

- -wavdir に音声コーパスの存在するディレクトリを指定します。
  上記サンプルの、網掛け部分（/16k）が追記部分です。
- -wavdir で指定したディレクトリは、list.txt ファイルに記述されている音声ファイルの上位ディレクトリ指定となります。
- -wavdir 指定したディレクトリに存在する音声ファイルは、データベース作成時のサンプリングレートとは異なっていてもかまいませんが、データベース作成に使用した音声ファイルと時間同期がとれた（ダウンサンプリング前の音声ファイルであるなど、時間情報が同じ）音声ファイルをご指定ください。

5．7 データベースロードエラー時の対応

作成したデータベース内にあらかじめ決められた音素（DBname¥data¥phdef.txt にて定義）がないと、データベースのロードに失敗します。

XIMERA GUI でデータベースをロードした際、ステータスバーに表示されるエラーを確認の上、xdata¥[speaker]¥seg¥[DBname]¥data¥phdefmap.txt に代替音素を追加してください。

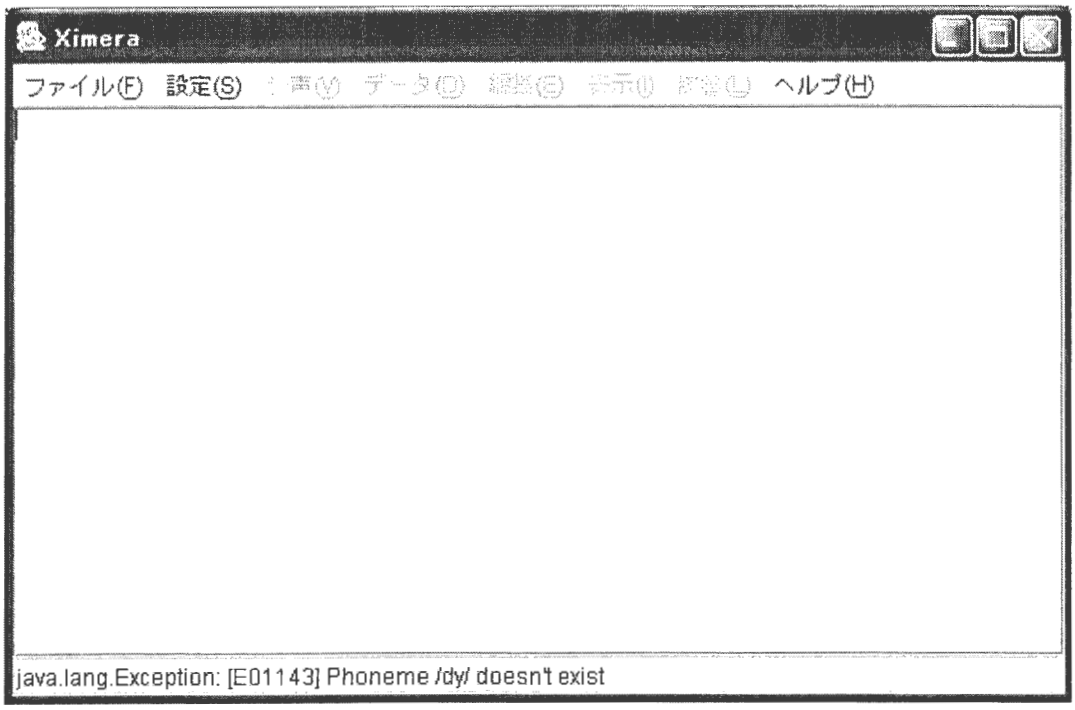


図 5．6 データベースロードエラー画面サンプル

phdefmap.txt ファイルのフォーマットは、下記のとおりです。

音素 -> 代替音素 件数

[DBname]¥data¥phdefmap.txt サンプル

```
……
tsU -> ts 50
dy -> d 1
```

網掛け部分は、データベースに「dy」が 1 件以下しか存在しない場合、「dy」を「d」に置き換えて、音声合成を行うことを定義しています。

5．8 合成エラー発生時（ポーズが存在しないデータベース）の対応

作成したデータベース内にポーズがない場合（音声ファイルが単語のみの場合など)、ポーズが挿入される文章を合成する際に以下のエラーが発生します。

エラー出力サンプル

```
FATAL ERROR: [E01111] no such phoneme [pau]
```

この場合、xdata¥[speaker]¥seg¥[DBname]¥data¥phdefmap.txt に代替音素を追加してください。

[DBname]¥data¥phdefmap.txt サンプル

```
……
tsU -> ts 50
pau -> sil 1
```

網掛け部分は、データベースに「pau」ラベルが 1 件以下しか存在しない場合、「pau」を「sil」に置き換えて、音声合成を行うことを定義しています。

６． F0・ラベル修正ツール（PIIKUN）使用方法

「F0・ラベル修正ツール」は「データベース作成ツール」により、自動的に作成した F0 の初期値とラベル位置の初期値を任意で修正する場合に使用します。

＜注意＞ラベルの文字列を変更・追加・削除することはできません。

このツールは、Javaで記述されていますのであらかじめJ2SE™ v1.4.2_04 をインストールした環境が必要です。

使用方法は、Windows 版・Linux 版ともに基本的に同じです。実際にお試しいただく際に必要なサンプルデータを以下にご用意しています。ご利用ください。

Ximera110¥sample¥piikun

6．1 入力ファイルの準備

「F0・ラベル修正ツール」を使用するには以下のデータファイルが必要です。

表 6．1 F0・ラベル修正入力ファイル

①音声ファイル	音声切り出しツールにより 1 文単位に分割したもの。 RIFF ヘッダー付 PCM ファイル。 16bit 量子化 モノラル。 サンプリングレートは 16kHz または 48kHz。	拡張子： 任意ファイル名.wav
②F0 ファイル	①を基にデータベース作成ツールにより作成されたもの。	拡張子： 任意ファイル名.xf0 《修正済のファイル=任意ファイル名.f0 がある場合は修正済ファイルが優先。》
③ラベルファイル	カナファイル（.kna）を基にデータベース作成ツールにより作成されたもの。	拡張子： 任意ファイル名.xmk 《修正済のファイル=任意ファイル名.mk がある場合は修正済ファイルが優先。》

＜注意＞

1. ①音声ファイル ②F0 ファイル ③ラベルファイル は同じディレクトリ内に置いてください。
2. 任意ファイル名が同じの ①音声ファイル ②F0 ファイル ③ラベルファイル が揃っていると、データファイルをロードして正常に作業画面が表示されます。
3. ①音声ファイルは、1 ファイルの長さが 16kHz で約 120 秒を超えるとメモリー不足のエラーメッセージが出る場合があります。

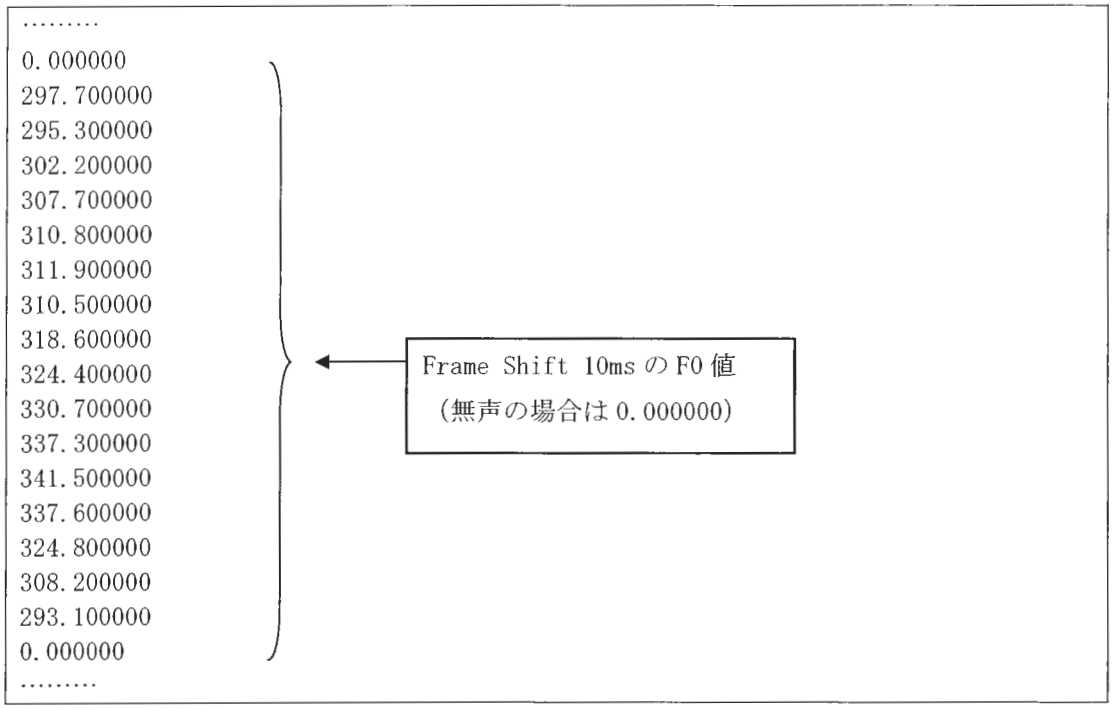


図 6.1 ②F0 ファイル（.xf0）の数値フォーマット例

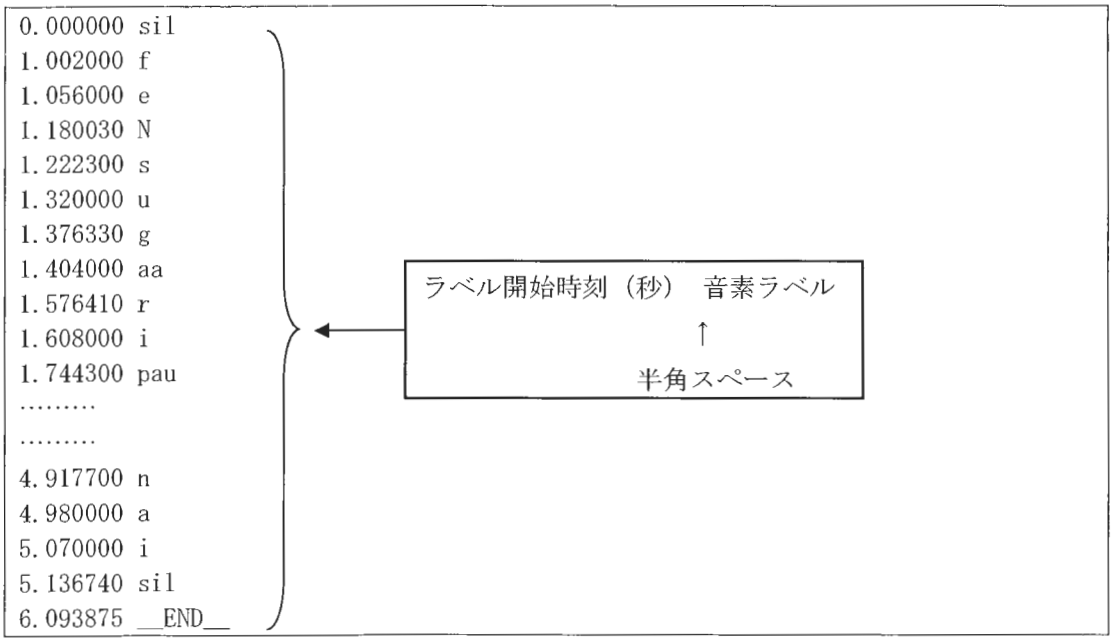


図 6.2 ③ラベルファイル（.xmk）の数値フォーマット例

６．２ 操作説明

６．２．１ 起動方法

●Windows 版

Ximera110¥src¥dbtool¥piikun¥piikun.bat を実行してください。

（java -Xmx512m -Xss10m -jar piikun.jar）

●Linux 版

Ximera110/src/dbtool/piikun/piikun.sh を実行してください。

（java -Xmx512m -Xss10m -jar piikun.jar）

以下の画面が表示されます。

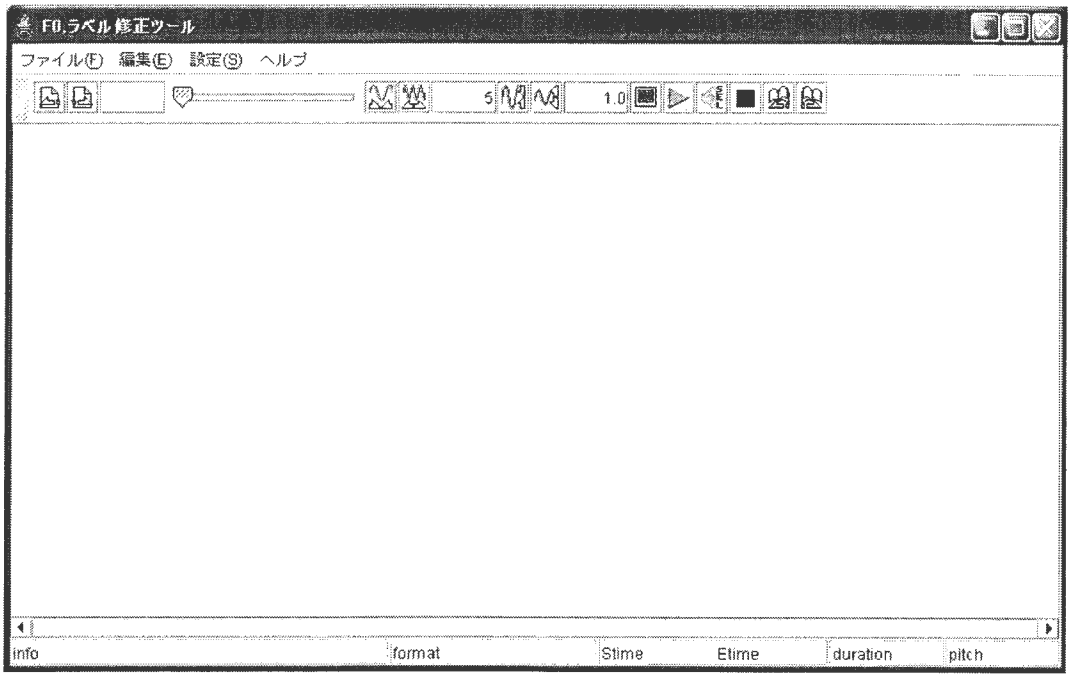


図 ６．３ F0・ラベル修正ツール メイン画面（初期画面）

６．２．２ データディレクトリの指定とデータ表示

F0・ラベル修正ツールを起動後まず、作業ディレクトリ（データディレクトリ）のパスを設定します。

F0・ラベル修正ツールメイン画面メニューバーの【ファイル（F)】〔作業ディレクトリ(Ctrl-O)〕メニューを実行し、表示されたディレクトリ選択画面でデータファイルがあるディレクトリを指定するとデータがロードされ作業画面が表示されます。

＜図 6.5 F0・ラベル修正ツール メイン画面（作業画面：1）＞を参照してください。

＜注意＞データファイルのフォーマットが違うと正常に表示されません。

＜表 6.1 入力ファイルの準備＞を参照してください。

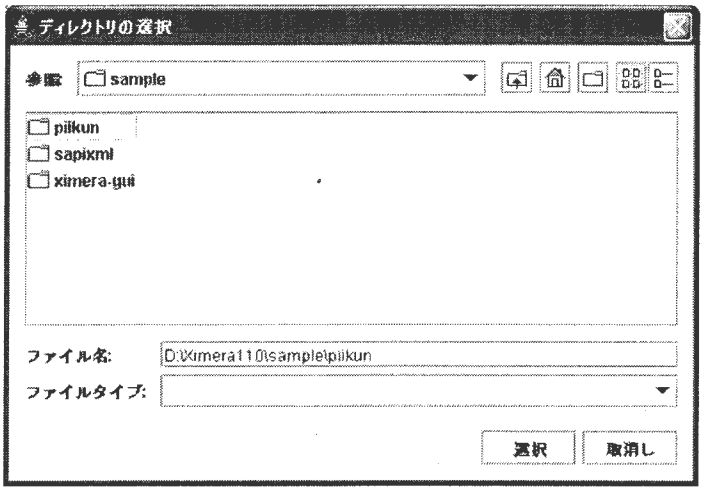


図 ６．４ 作業ディレクトリの指定画面

6．2．3 コマンドの機能と表示説明

以下の、図 6.5 の項目ごとに機能と表示を説明します。

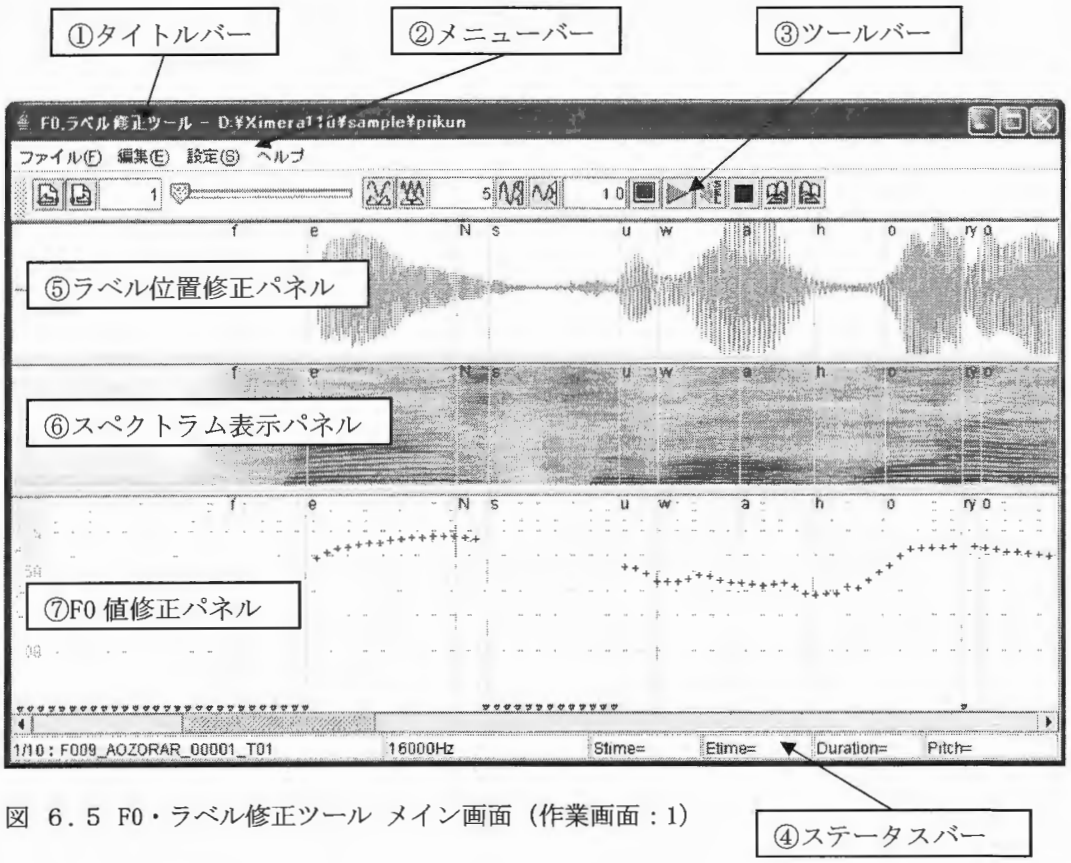


図 6．5 F0・ラベル修正ツール メイン画面（作業画面：1）

① タイトルバー

現在の作業ディレクトリ（データディレクトリ）が表示されます。

② メニューバー

表 6．2 【ファイル (F)】機能概要

プルダウンメニュー	機能概要
作業ディレクトリの指定 (Ctrl-O)	データファイルがあるディレクトリを指定します。
保存 (Ctrl-S)	F0 値・ラベル位置の初期値変更の有無にかかわらずデータファイルを保存します。 ロードした初期値のデータファイルとは別に、“ロードしたデータファイル名.f0”

	“ロードしたデータファイル名.mk”の２つのファイルを同時に作成します。 保存するごとに、この２つのファイルを上書き保存します。
次のデータへ移動（Ctrl-N）	同じディレクトリ内にある次のデータファイルをロードします。
前のデータへ移動（Ctrl-P）	同じディレクトリ内にある前のデータファイルをロードします。
終了（Ctrl-X）	F0・ラベル修正ツールを終了します。

表 ６．３ 【編集（E)】機能概要

プルダウンメニュー	機能概要
UNDO	直前に実行した操作を取り消して、元の状態に戻します。 （F0 値・ラベル位置の初期値修正のみ有効です。）
REDO	UNDO 機能を使って取り消した操作を再度やり直します。

表 ６．４ 【設定（S)】機能概要

プルダウンメニュー	機能概要
ピッチ表示範囲の設定	F0 値の表示範囲を設定します。 ＜図 6.6 ピッチ表示範囲の設定画面＞を参照してください。 値を変更すると＜図 6.5 ⑦F0 値修正パネル＞の左端のスケールが変わります。

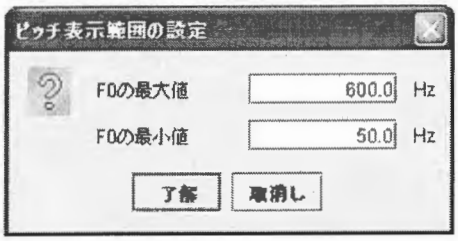


図 6.6 ピッチ表示範囲の設定画面（デフォルト値）

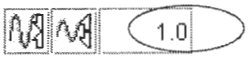
表 6.5 【ヘルプ】機能概要

プルダウンメニュー	機能概要
バージョン	F0・ラベル修正ツールのバージョン情報を表示します。

③ ツールバー

表 6.6 ツールバー機能概要

アイコンメニュー	機能概要
	同じディレクトリ内にある前のデータファイルをロードします。
	同じディレクトリ内にある次のデータファイルをロードします。
 現在ロードしているファイル：1	同じディレクトリ内にある任意のファイルにジャンプしてロードします。 （ファイル昇順番号を入力してください。）
 デフォルト値：5	＜⑤ラベル位置修正パネル＞ ＜⑥スペクトラム表示パネル＞ ＜⑦F0 値修正パネル＞ の時間軸方向の表示サイズを変更します。ボタンを動かすか、数字を入力します。 表示される数字は、標準表示サイズを 1 とした場合の倍数（整数）です。 上限は 100 倍です。
	現在表示されているパネルのサイズを 1/2 倍にします。表示サイズの数字は整数のみの

	ため、割り切れない場合は四捨五入して表示します。
	現在表示されているパネルのサイズを 2 倍にします。
 デフォルト値：1.0	音声波形の振幅倍率の表示を変更します。
	現在表示されている音声波形の振幅倍率を 1/2 倍にします。
	現在表示されている音声波形の振幅倍率を 2 倍にします。
	アイコンを押下すると <⑥スペクトラム表示パネル>を 非表示にします。
	アイコンを押下すると <⑥スペクトラム表示パネル>を 表示にします。
	表示している音声ファイル全体を再生します。
	表示している音声ファイルの一部を範囲選択して再生した音声を、もう一度再生します。
	再生中の音声を途中で中止します。
	表示されているパネルの前の部分をページごとに表示します。
	表示されているパネルの後ろの部分をページごとに表示します。

④　ステータスバー

各種データ情報を表示します。

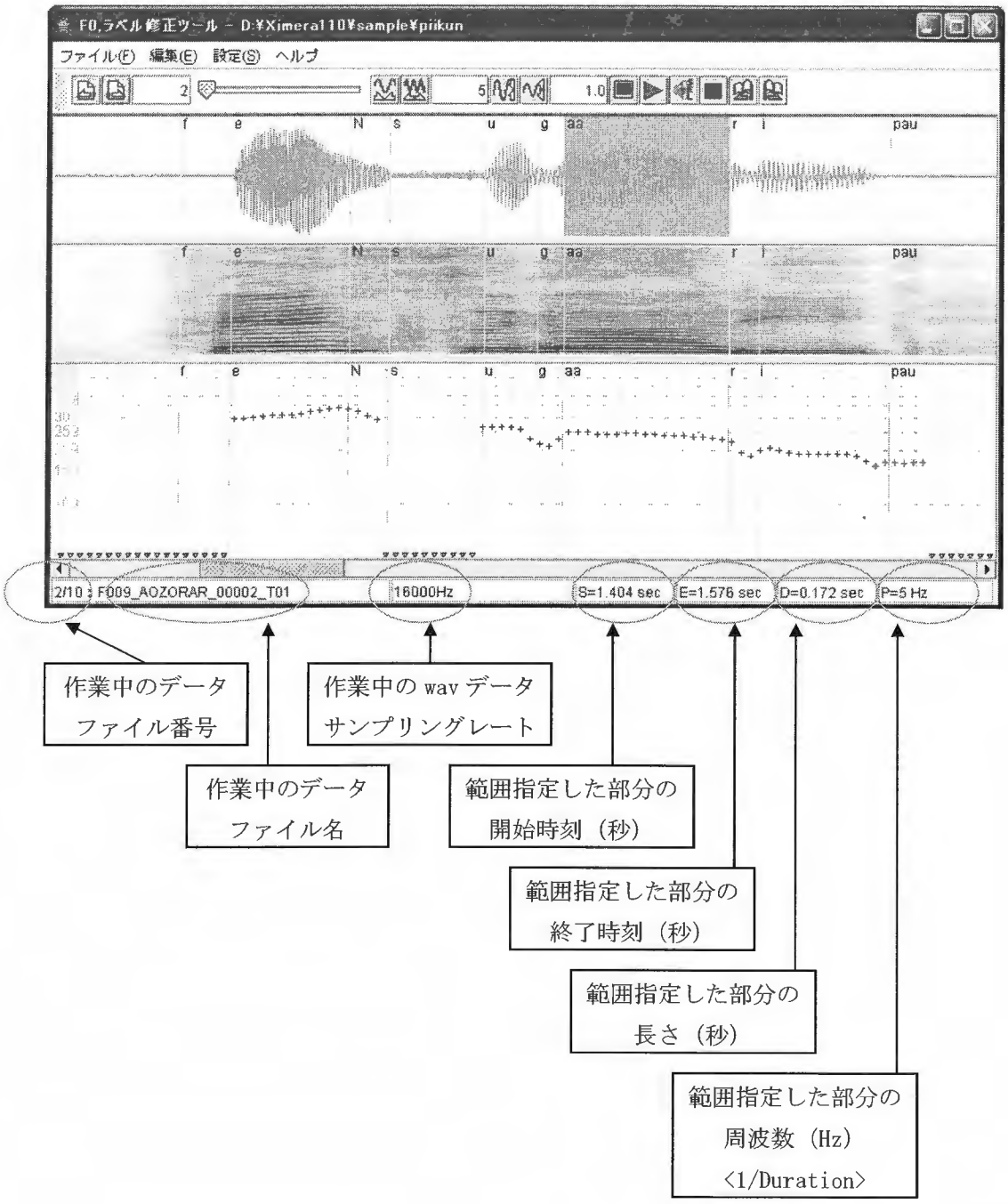


図　6．7　F0・ラベル修正ツール　メイン画面（作業画面：2）

⑤　ラベル位置修正パネル

◎範囲指定した部分の音声を再生します。

・１音素区間の音声を再生する場合
　再生したい音素区間にカーソルを合わせて、マウス右ボタンをダブルクリックします。
　範囲選択された部分がブルーに変わると同時に音声が再生されます。
　もう一度聞く場合は、ツールバーのアイコンメニューで再生します。
　＜表 6.6 ツールバー機能概要＞を参照してください。

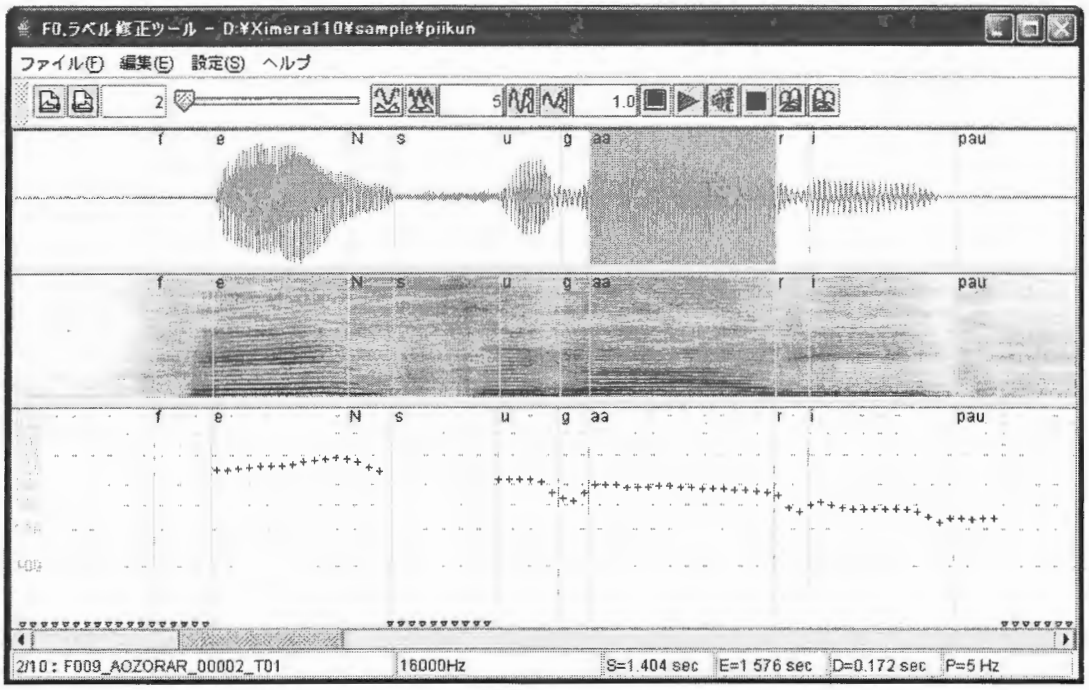


図 6.8 範囲選択部分の音声の再生（１音素区間）

・任意区間の音声を再生する場合

再生したい任意区間を、マウス右ボタンでドラッグします。

範囲選択された部分がブルーに変わると同時に音声が再生されます。

もう一度聞く場合は、ツールバーのアイコンメニューで再生します。

＜表 6.6 ツールバー機能概要＞を参照してください。

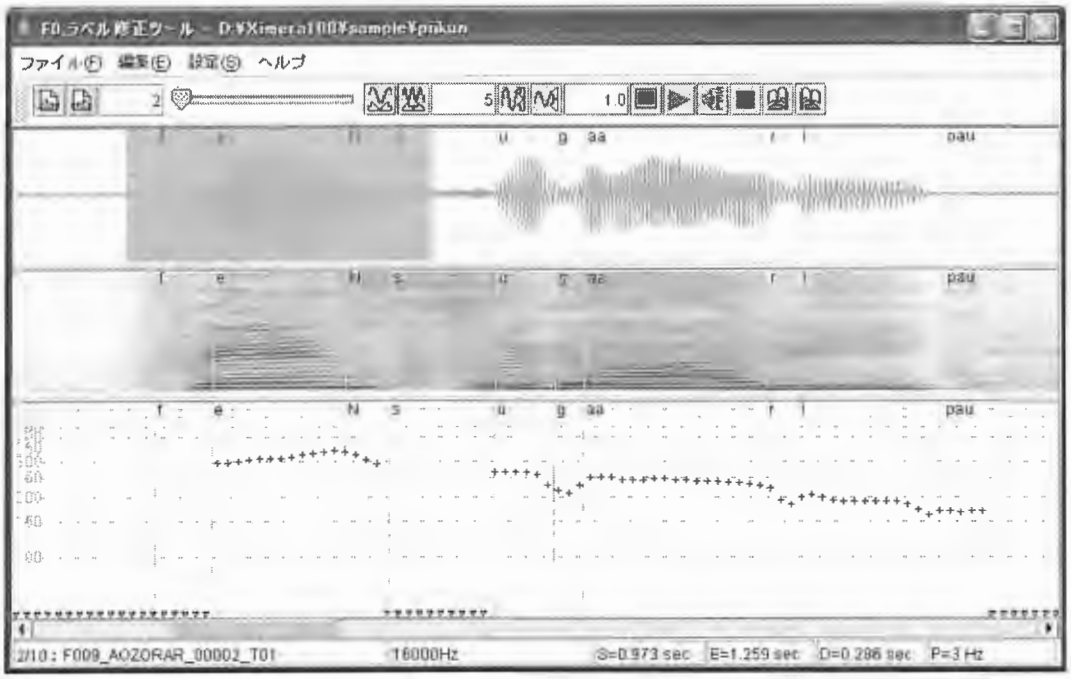


図 6．9 範囲選択部分の音声の再生（任意区間）

◎ラベル位置を修正します。

＜修正例＞

“b”のラベル位置を修正したい場合、動かしたいラベル位置（線）にカーソルを合わせると、表示がデフォルトカーソルからハンドカーソルに変わります。
この時点でマウス左ボタンでドラッグし始めると線がブルーになり、修正したい位置まで移動することができます。
＜図 6.10 ラベル位置修正例＞を参照してください。

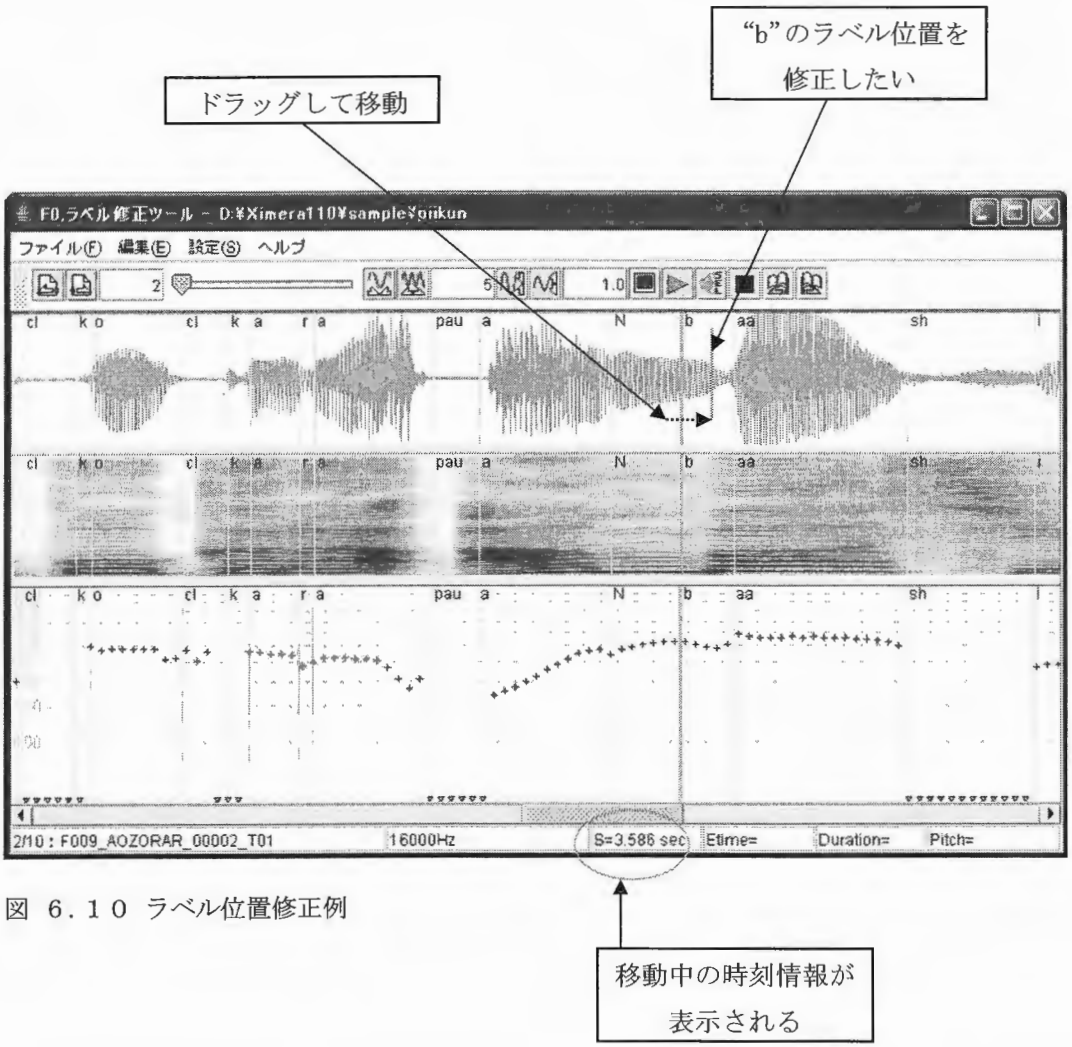


図 6．10 ラベル位置修正例

⑥　スペクトラム表示パネル

このパネルでの機能はありません。

⑦　F0 値修正パネル

◎F0 値を修正します。

＜修正例-1＞

“cl”の F0 値を修正したい場合、動かしたいポイントをマウス左ボタンでドラッグしてブルーのラインで囲みます。

＜図 6.11 F0 値修正例-1＞を参照してください。

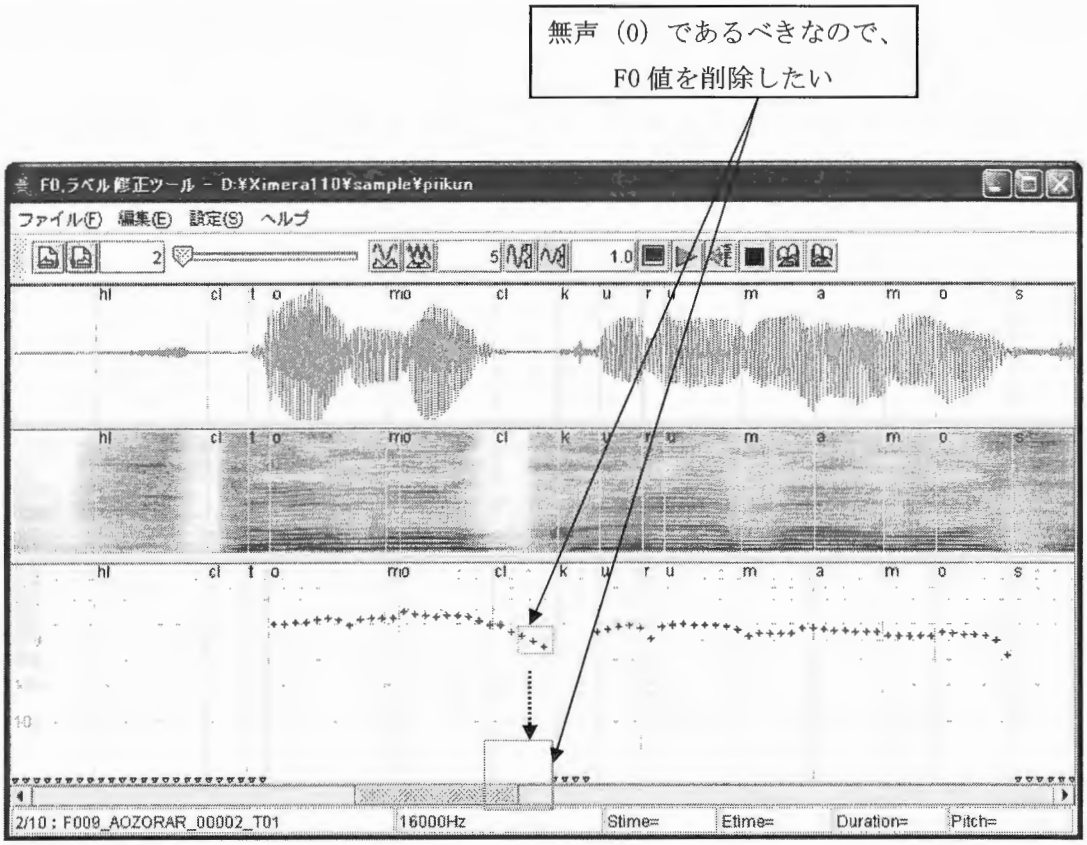


図 6.11 F0 値修正例-1

動かしたいポイントをマウス左ボタンでドラッグしてブルーのラインで囲むと、指定したポイントの色がレッドからグリーンに変わります。

ポイントにカーソルを合わせると表示がデフォルトカーソルからハンドカーソルに変わります。

マウス右ボタンでポップアップメニューを表示し〔選択データを削除〕を押下すると、F0 値が“0.0”になります。

<図 6.12 F0 値修正例-1>を参照してください。

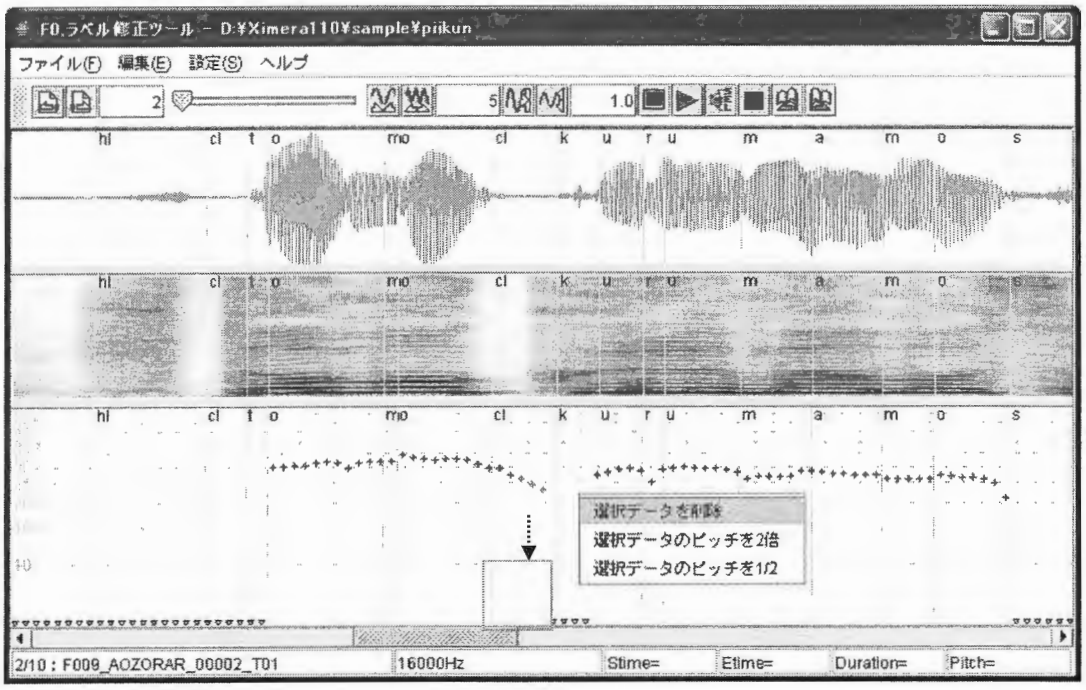


図 6.12 F0 値修正例-1

表 6.7 F0 値修正パネル ポップアップメニュー機能概要

ポップアップメニュー	機能概要
選択データを削除	F0 値を“0.0”にします。
選択データのピッチを 2 倍	選択データの現在の F0 値を 2 倍にします。
選択データのピッチを 1/2	選択データの現在の F0 値を 1/2 倍にします。

＜修正例-2＞

“oo”のF0値を修正したい場合、動かしたいポイントをマウス左ボタンでドラッグしてブルーのラインで囲むと指定したポイントの色がレッドからグリーンに変わります。カーソルを合わせると、カーソルの表示がデフォルトカーソルからハンドカーソルに変わります。

この時点でマウス左ボタンでドラッグするとポイントの色がブルーになり、修正したい位置まで移動することができます。

＜図 6.13 F0値修正例-2＞を参照してください。

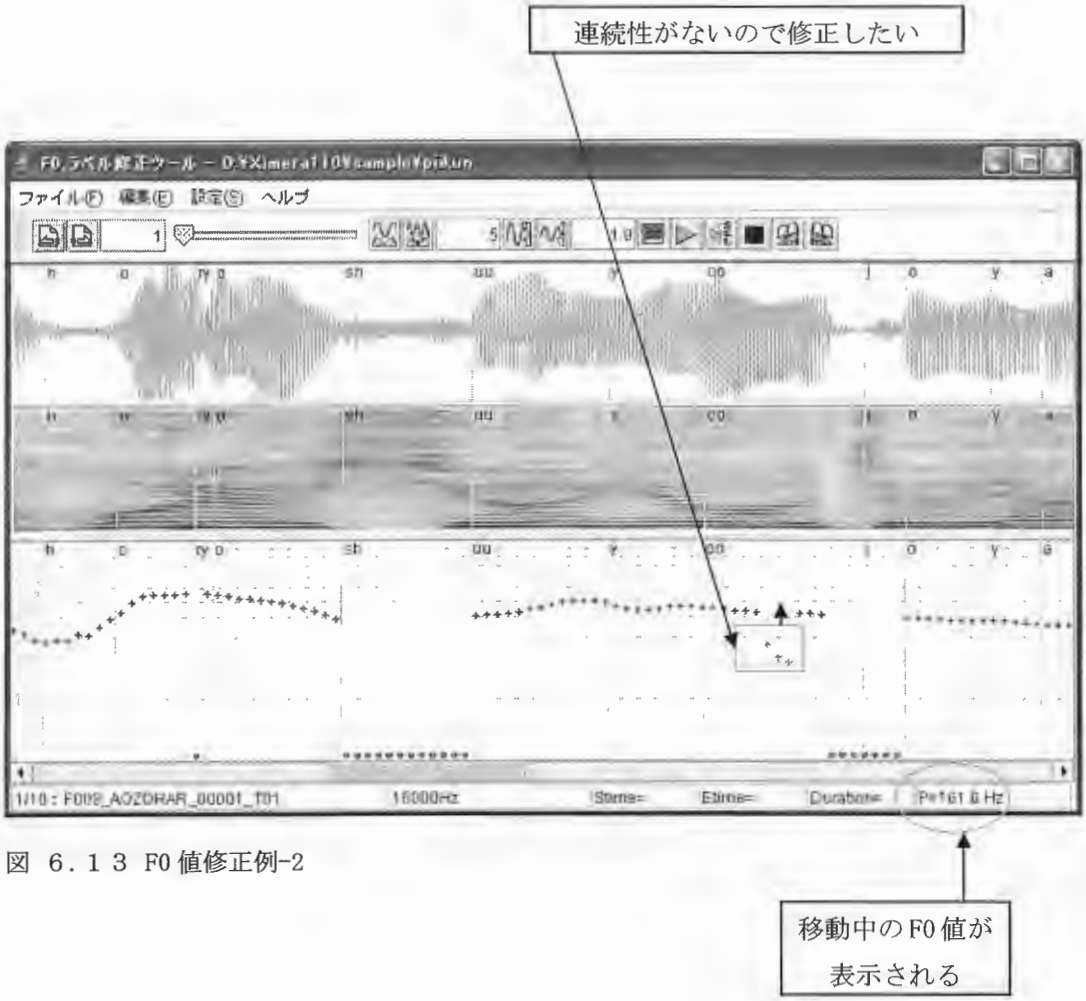


図 6.13 F0値修正例-2

＜修正例-3＞

F0 値を修正するために、波形から周波数を計算します。

音声波形の振幅倍率を拡大し、波形周期の一番高い 2 つピークをマウス右ボタンでドラッグし指定します。計算された F0 値がステータスバーに表示されます。

表示された F0 値まで移動します。

＜図 6.14 F0 値修正例-3＞を参照してください。

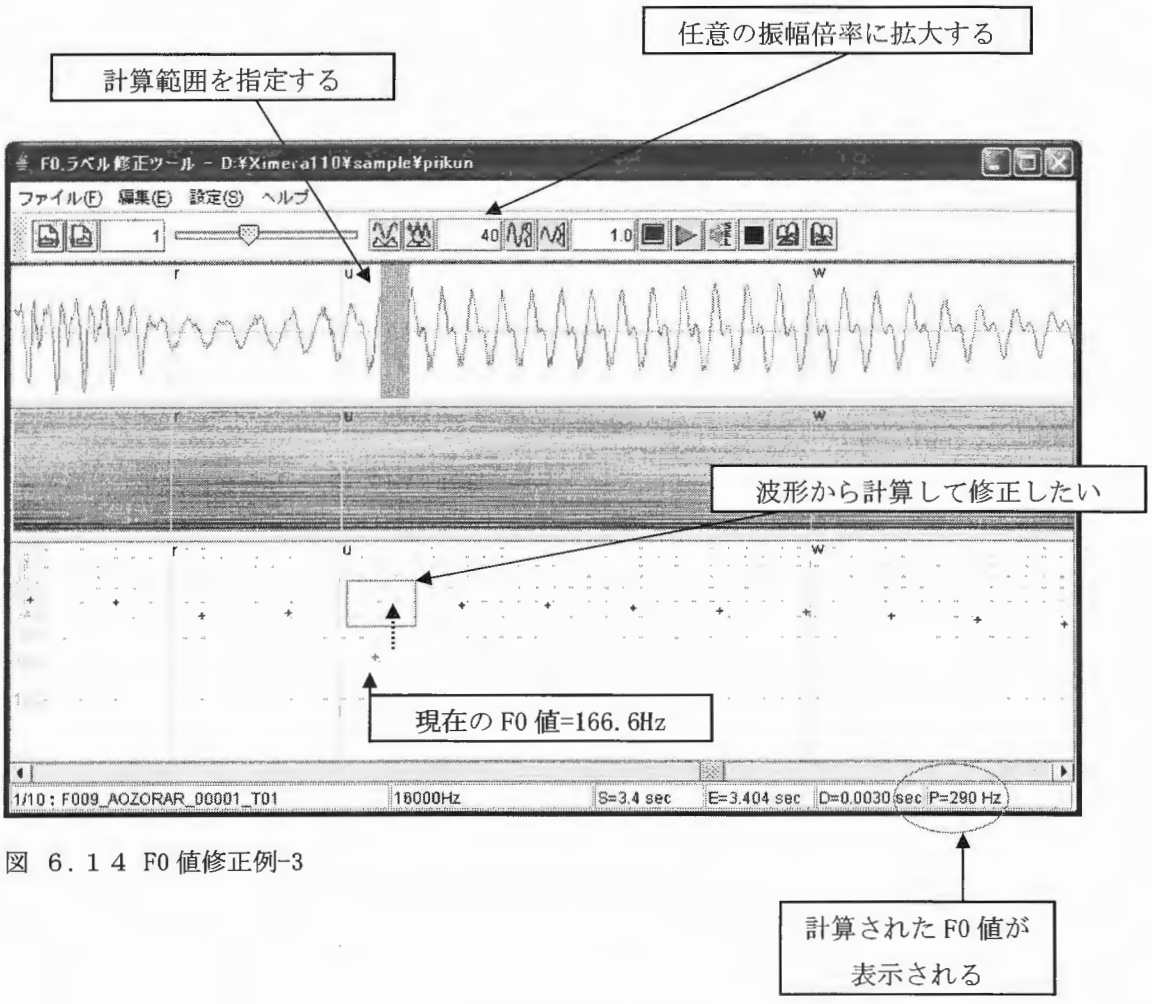


図 6.14 F0 値修正例-3

6．3 データファイルのロード優先順位

・「F0・ラベル修正ツール」で使用するデータファイルは、wav ファイルが“基準ファイル”になります。F0 ファイル・ラベルファイルが揃っていても wav ファイルがないとロードしません。

・作業中のデータファイル番号（＜図 6.7 F0・ラベル修正ツール メイン画面＞を参照してください。）の“2/10”の“10”は作業ディレクトリにある全ファイル数を指します。この全ファイル数とは wav ファイル数（拡張子＝.wav）となります。

・作業ディレクトリに、F0 の初期値（.xf0 ファイル）・ラベル位置の初期値（.xmk ファイル）と、F0 修正値（.f0 ファイル）とラベル位置修正値（.mk ファイル）が混在している場合は、修正値をロードして画面に表示します。

7． HTS 学習

HTS 学習 GUI を使用し、XIMERA の韻律予測に使用する話者固有の HMM を作成します。

7．1 入力ファイル

HTS 学習 GUI で使用するファイルは以下のとおりです。

表 7．1 HTS 学習入力ファイル

音声ファイル	ファイルの拡張子は、wav RIFF ヘッダー付 サンプリングレートは 16kHz
言語韻律情報ファイル	ファイル名は、（音声ファイル名）.pli 漢字コードは、SJIS、EUC のいずれかを使用
ラベルファイル	ファイル名は、（音声ファイル名）.xmk または、（音声ファイル名).mk（修正済み） 修正済みラベルファイルがある場合は、修正済みを使用
F0 ファイル	ファイル名は、（音声ファイル名）.xf0 または、（音声ファイル名).f0（修正済み） 修正済み F0 ファイルがある場合は、修正済みを使用

7．2 入力ファイルの作成方法

HTS 学習の入力ファイルを下記手順で作成します。「1.4 HMM 作成フロー」をご参照ください。

7．2．1 言語韻律情報の修正

HTS 学習の対象となるサブコーパスについて、音声切り出しツールで出力された言語韻律情報ファイル(*.pli)を言語韻律情報エディタ（PLEd）もしくは、テキストエディタで正しく修正します。

読み付与間違いがある場合は、言語韻律情報ファイルを修正してください。また、品詞やアクセント区切りが間違っている場合も修正を行ってください。

7．2．2 カナファイルの作成（pli2kana）

pli2kana ツールを使用し、修正済み言語韻律情報ファイル（*.pli）からカナファイル(*.kna)を作成します。

使用方法は、

pli2kana ［データパス］［言語韻律情報ファイル(in)］［カナファイル(out)］

使用サンプル

```
> pli2kna  /home/ximera/Ximera110/xdata  A01.pli  A01.kna
```

上記例では、言語韻律情報ファイル A01.pli から、カナファイル A01.kna を作成します。

7．2．3 ラベルファイル、F0 ファイルの作成

pli2kana で作成したカナファイルを音声ファイルと同じディレクトリに配置します。データベース作成 GUI で対象となるサブコーパスを選択し、align、f0 の工程を実行してください。

音声ファイルと同じディレクトリに、ラベルファイル（*.xmk)、F0 ファイル(*.xf0)が作成されます。

7．2．4 ラベルファイル、F0 ファイルの修正

データベース作成 GUI で作成した F0 ファイル（*.xf0）、ラベルファイル（*.xmk）、を修正し、修正済み F0 ファイル（*.f0）、修正済みラベルファイル（*.mk）を作成します。

この作業は、省略可能です。

F0 ファイル、ラベルファイルの修正方法については、「6. F0・ラベル修正ツール（PIIKUN）使用方法」をご参照ください。

7．3 入力ファイルをディレクトリに配置

以下の入力ファイルを同一のディレクトリに配置します。

- 音声ファイル(*.wav)
- 言語韻律情報ファイル (*.pli)
- ラベルファイル(*.xmk) または、修正済みラベルファイル(*.mk)
  (両方ある場合は、修正済みを優先して使用します。)
- F0 ファイル(*.xf0)または、修正済み F0 ファイル(*.f0)
  (両方ある場合は、修正済みを優先して使用します。)

7．4 HTS 学習 GUI

7．4．1 HTS 学習 GUI の起動

Windows の場合

Ximera110¥src¥dbtool¥training-gui¥HMM.bat

Linux の場合

Ximera110/src/dbtool/training-gui/HMM.sh

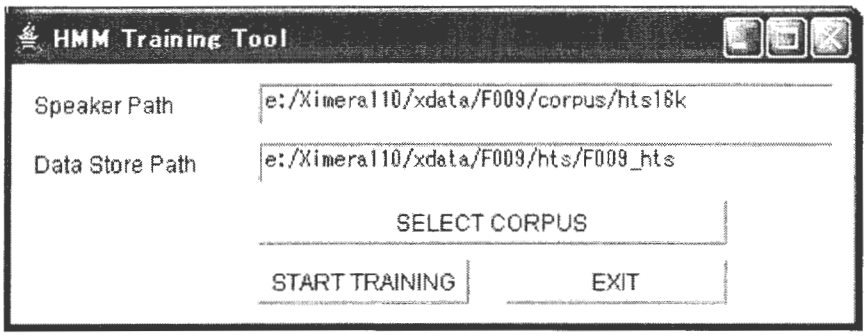


図 7．1 HTS 学習初期画面サンプル

7．4．2 HTS 学習の実行

下記を入力します。

Speaker Path　　入力ファイルが存在する上位ディレクトリ

Data Store Path　　HTS 学習結果を出力するディレクトリ

サブコーパスの選択

SELECT CORPUS ボタンを押下し、HTS 学習に使用するサブコーパスを選択します。

実行

START TRAINING ボタンを押下します。

※ およその実行時間は次のとおりです。

使用マシン：　CPU Pentium(R) 4 CPU 2.40GHz

メモリ　2GB

コーパス：　文章 503 文（16kHz の音声ファイル約 35 分）

実行時間：　約 12 時間

7．4．3 HTS 学習結果の確認

HTS 学習が完了すれば、[Data Store Path]ディレクトリに HTS 学習結果ファイルが作成されています。

XIMERA エンジンから使用するのは、[Data Store Path]ディレクトリ直下のファイルのみです。（サブディレクトリ以下は不要です。）

7．5 HTS 学習エラー時の対応

コーパスに含まれる音素が、HTS 学習時に 1 件も使用できなかった場合（中間ファイル作成に失敗する等）、HTS 学習起動画面に下記のようなエラーが表示されます。

HTS 学習エラー表示サンプル

```
Failed : Failed to create initial model. See no_phone.txt
(Press Enter to terminate this system.)
```

この場合、[Data Store Path]¥no_phone.txt を参照し、下記いずれかを行って下さい。

- 学習に不足した音素の削除
  学習に不足した音素が含まれる音声ファイルをすべて削除
  （下記例では、”ny”音素が含まれるファイルを削除します。）
  再度 HTS 学習を実行

- 学習に不足した音素の追加
  学習に不足した音素を含む音声ファイルを追加
  再度 HTS 学習を実行

[Data Store Path]¥no_phone.txt サンプル

```
ny : Failed to create a initial model.
```

※ 上記サンプルは、音素「ny」が 1 件も学習に使用できなかったことを示しています。
※ 「：」以降はコメントです。

XIMERA（Ver1.1）

ユーザーズマニュアル

= SAPI =

２００４年６月３０日

株式会社 国際電気通信基礎技術研究所
Advanced Telecommunications Research Institute International

ATR

目次

はじめに

Microsoft Speech API (以下「SAPI」と記す) とは、音声アプリケーションのための Windows 標準インターフェース群です。

XIMERA は SAPI に準拠することを目的としており、SAPI に規定されているほとんどの機能を満たしています。
よって、音声アプリケーションから SAPI インターフェースを介して、XIMERA を使用することが可能です。

また、Windows XPには、SAPIが標準で組み込まれていますので、Microsoft Speech SDK5.1[1] のインストールの必要はありません。
ただし、SAPI のヘルプを参照する場合や、Compliance Tool を実行する場合は、インストールが必要となります。

[1] Microsoft Speech SDK5.1 は、http://www.microsoft.com/speech/download/sdk51/ からダウンロード可能です。

1．SAPI への登録と設定

1．1 SAPI への登録

SAPI で XIMERA を使用するには、DLL（COM オブジェクト）の登録と、データベースの登録を行います。

1．1．1 DLL（COM オブジェクト）の登録

レジストリに Ximera.dll を登録します。

```
> cd c:¥Ximera110¥bin¥windows
> regsvr32  Ximera.dll
```

※ 網掛け部分はインストールしたディレクトリを指定してください。
※ 既に登録済みの DLL がある場合は、登録済みの DLL が存在するディレクトリで削除後、実行してください。（「1.1.2 DLL の削除」参照）

＜注意＞
ximera-sapi モジュールをコンパイルすると、作成した Ximera.dll が自動的に再登録されます。明示的に DLL の削除、登録を行う必要はありません。

1．1．2 DLL（COM オブジェクト）の削除

登録した DLL をレジストリから削除するには、下記コマンドを実行してください。

```
> regsvr32  /u  Ximera.dll
```

１．１．３ データベースの登録

SAPIからデータベースを使用するために、データベースの登録を行います。
登録は、話者・HMM・データベースの組み合わせで必要な回数だけ行います。

SetupXimera.exeの -installオプションを使用して登録を行います。
installのオプションには、データパス・話者・HMM・データベースを指定します。

```
> cd c:¥Ximera110¥bin¥windows
> SetupXimera.exe  -install  c:¥Ximera110¥xdata  F009 F009_hts F009_503
install
Reading Database ...
create reg keys...Success
```

※ 網掛け部分はインストールしたディレクトリを指定してください。
※ 以前に登録したデータベースがあれば、削除してください。(「1.1.4 登録したデータベースの削除」参照)
※ 画面に上記メッセージが表示すれば、完了です。
※ 上記は、データベースF009_503を登録する場合の例です。
※ データベース F009 を使用するには、データベース名に F009 を指定し、再度SetupXimera.exe を実行してください。

エンジンへの登録が完了すると、レジストリの
HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Speech¥Voices¥Token
以下に登録したデータベースの情報が書き込まれます。詳しくは、Microsoft Speech SDK5.1のヘルプをご参照ください。

１．１．４ 登録したデータベースの削除

登録したデータベースを削除するには、setupximera.exeの -uninstallオプションを使用します。uninstallのオプションには、話者・HMM・データベースを指定します。

```
> cd Ximera110¥bin¥windows
> SetupXimera.exe  -uninstall  F009 F009_hts F009_503
```

１．２ 合成エンジンの設定

データベースの登録が完了すると、SAPI より登録したデータベースが使用可能になります。よって、コントロールパネルより合成エンジンの設定が可能です。

登録したデータベースは、話者、HMM、データベースの３つがハイフンでつながれた名前で表示されます。

（例）コントロールパネル → サウンド、音声、およびオーディオ デバイス → 音声認識で設定画面を表示し、音声の選択を行って下さい。

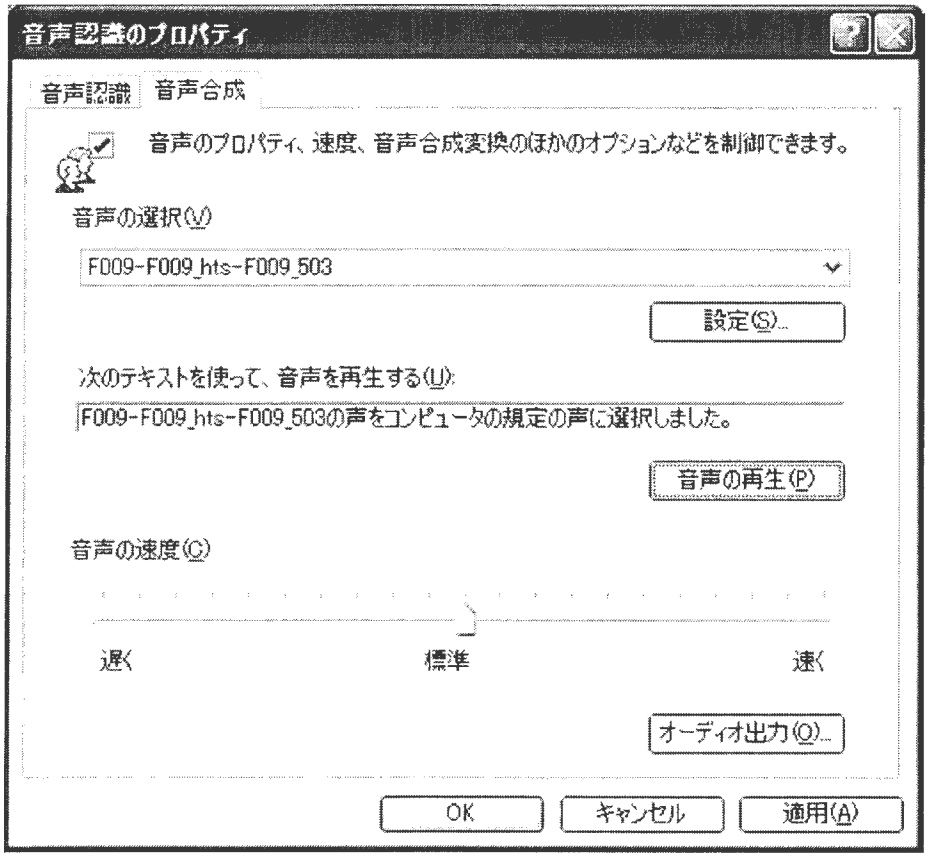


図 １．１ コントロールパネルの設定画面

１．３ ユーザー辞書

XIMERA では、ユーザー辞書を使用することができます。
辞書を登録するには、コントロールパネルの設定画面（図 1.1 参照）の設定ボタンを押下し、Ximera 辞書設定画面で表記と読み（カタカナ）を入力後、追加ボタンを押下してください。

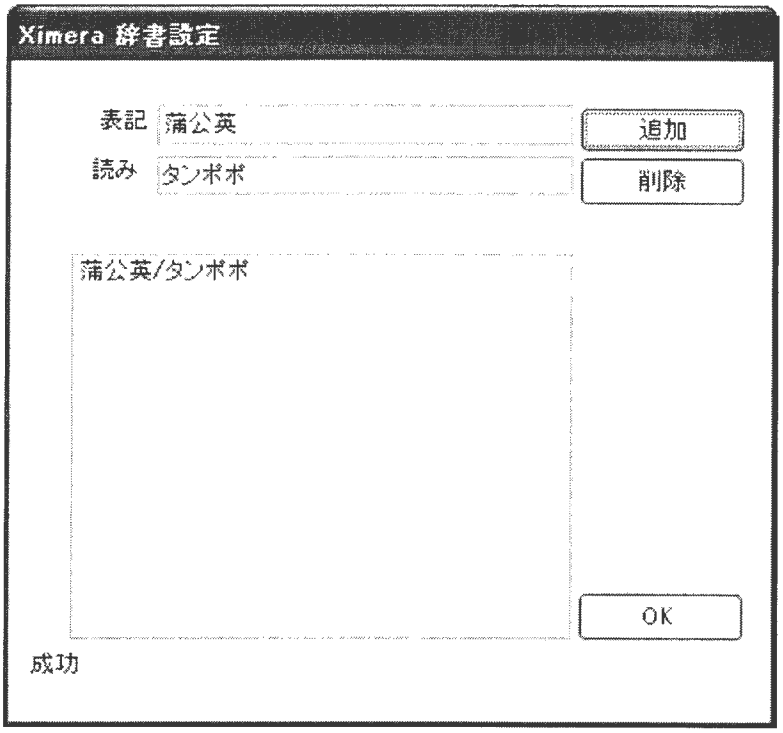


図 １．２ 辞書設定画面サンプル

登録された辞書は、SAPI より XIMERA を使用する際、使用されます。

2． XML Schema タグ

XIMERA が対応する XML タグは、表 2.1 XML Schema タグのとおりです。
CONTEXT, PARTOFSP, SPELL タグには対応しておりません。
タグの詳細説明については、Microsoft Speech SDK5.1 のヘルプをご参照ください。

＜XIMERA Version1.1 未対応タグについて＞
XIMERA Version1.1 では、PITCH、RATE の 2 つのタグに対応しておりません。これらのタグを使用した文章の合成は可能ですが、タグが効いていない音声となります。

表 2.1 XML Schema タグ

Tag	attributes	有効／無効	Ver1.1 対応	内容
BOOKMARK	MARK			ブックマークの挿入
CONTEXT	ID	XIMERA では無効		データ内容に関する情報を記述 日付・時間の記述など
EMPH				強調
PARTOFSP	PART	XIMERA では無効		品詞を指定し、特別の発音を指定
PITCH	MIDDLE ABSMIDDLE		未対応	ピッチの指定（−10 ～ +10)
PRON	SYM			発音（読み）をカナで指定
RATE	SPEED ABSSPEED		未対応	発音スピード指定（−10 ～ +10)
SILENCE	MSEC			無音の指定（単位ミリ秒)
SPELL		XIMERA では無効		つづりよみを指定
VOICE	REQUIRED OPTIONAL			話者の指定
VOLUME	LEVEL			音量の指定（0 ～ 100)

サンプルの XML は、Ximera110¥sample¥sapixml にあります。ご参照ください。
また、Microsoft Speech SDK5.1 付属のサンプルアプリケーションに TTSApp があります。
VB6.0 日本語版でコンパイルすると日本語が扱えるようになり、XML タグのテスト等、SAPI の動作確認を容易に行うことができます。

3. TTS Compliance Tool

Microsoft Speech SDK5.1に付属のCompliance Toolは、音声合成エンジンまたは音声認識エンジンがSAPIに対応しているかをチェックするためのツールです。
XIMERAをCompliance Toolにかける方法と、その結果について説明します。

3.1 Compliance Tool実行方法

Compliance Toolを実行するには、あらかじめ、Microsoft Speech SDK5.1をインストールしたマシンをご用意ください。また、Compliance Toolの詳細な使用方法は、Microsoft Speech SDK5.1のヘルプをご参照ください。

＜実行例＞
(1) コントロールパネルより、合成エンジンの設定を行います。
XIMERAのデータベースを選択してください。

(2) Compliance Toolを起動
[SAPIインストールディレクトリ]¥Tools¥Comp¥Bin¥spcomp.exeを起動します。
スタートメニューからの起動も可能です。

(3) DLLのロード
Fileメニューから、Load Test DLLを選択します。
[SAPIインストールディレクトリ]¥Tools¥Comp¥Bin¥ttscomp.dllをご使用ください。

(4) テストセットの選択
TestsメニューからSelect Testsを選択し、実行したいテストセットを設定します。

(5) 実行
TestsメニューからRun Testsを実行します。

(6) 結果表示
画面に結果が表示されます。

3.2 Compliance Tool 実行結果

Compliance Tool のテスト結果は以下のとおりです。

表 3.1 Compliance Tool 結果

Required Test			結果(*1)
	ISpTTSEngine		
1	Speak	合成音声の作成	○
2	Skip	文章単位のスキップ処理	○(*3)
3	GetOutputFormat	ファイルフォーマットを返す	○
4	SetRate	スピード設定	Ver1.1 未(*3)
5	SetVolume	音量設定	○(*3)
	Eventing		
6	Check SAPI required Events	必須イベントチェック	○
	TTS XML Markup		
7	Bookmark	ブックマークイベント処理	○
8	Silence	無音の出力	○
9	Spell	綴り読み	○(*2)
10	Pronounce	読み指定（カナ記述）	○
11	Rate	スピード	Ver1.1 未
12	Volume	音量	○
13	Pitch	高さ	Ver1.1 未
14	Non-SAPI tags	エンジン固有のタグ	なし
15	Context	データ内容の記述（日付など）	○(*2)
	Real Time Rate/Vol Tests		
16	Real time rate change	リアルタイムスピード変更	Ver1.1 未(*3)
17	Real time volume change	リアルタイム音量変更	○(*3)
	Audio State Tests		
18	Speak Stop	読み上げ中断	○
19	Speak Destroy	読み上げ破棄	○
	Lexicon Tests		
20	User Lexicon Test	ユーザー辞書の登録削除	○
21	App Lexicon Test	アプリケーション辞書の登録・削除	○
	Multiple Instance Test		
22	Multiple-Instance Test	SAPI から多数呼び出しを同時に扱う	×(*4)
Option Test			
	Features		
1	Emph	強調処理	○
2	Phoneme & Viseme Events	音素、Viseme 単位の Event	○
3	PartOfSp	品詞情報による処理	×

(*1) 結果欄は、Compliance Tool の結果が「PASS」「SUPPORTED」の場合「○」、「FAIL」「UNSUPPORTED」の場合「×」で記述しています。
「Ver1.1未対応」と記述しているものは、XIMERA Version1.1では2つのタグに対応していないため、結果が「FAIL」となります。

(*2) TTS XML Markup の Spell と Context は、Compliance Tool の結果が「PASS」になりますが、XIMERA では処理しておりません。

(*3) Skip 処理やリアルタイム処理は、1文単位（句点単位）で処理しています。

(*4) XIMERA Version1.1では未対応のタグが存在するため、「FAIL」になります。

＜テスト用 DLL の変更点＞

Compliance Tool に使用した DLL は、Microsoft Speech SDK5.1 に付属の TTS 用 DLL (ttscomp.dll)を一部変更し、使用しています。変更点は次のとおりです。

- Skip テスト用の文章（STRING1）を句点で区切った文章に変更しています。句点単位に1文としているためです。