電話端末用フロントエンドの仕様　Ver.1.0

仕様作成者：青木 一峰、小林 剛典
Last Update: 2002/09/30

1. はじめに

電話端末用フロントエンドは，図 1 に示すように，携帯電話（一般電話）と電話サーバから構成される.ユーザが使用する電話端末としては､市販の携帯電話および一般電話を想定する｡また､電話サーバには音声および DTMF の処理が可能な音声処理ボードが組み込まれていることを前提とする。

本仕様では、電話サーバが処理できる入出力モダリティを規定する。

ユーザは、携帯電話（一般電話）を用いて電話サーバにアクセスすることにより、様々な対話サービスを利用できる。

本フロントエンドで利用できる入力モダリティは、

- 音声
- DTMF
- タイマー

であり、出力モダリティは、

- 音声
  - 合成音声
  - 録音音声
- トーン
- タイマー

である。

以下に各入出力モダリティについての詳細を述べる。

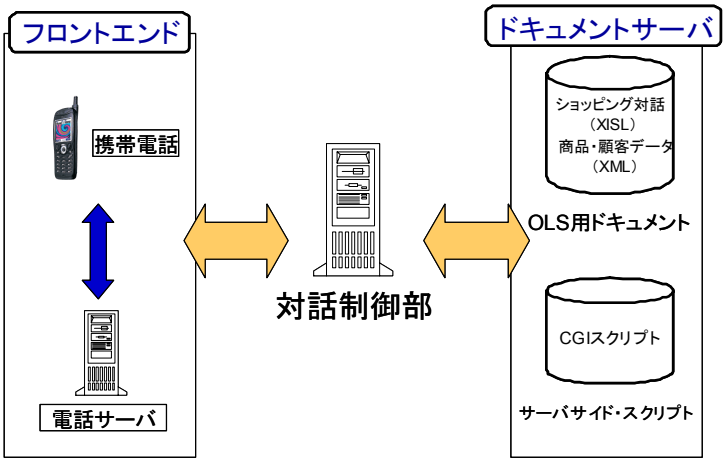

図 1：フロントエンド概要

2. 入力モダリティ

2.1 音声

2.1.1 音声認識完了イベント

▼ 機能

- 指定された音声認識語彙文法にマッチする発話パタンが入力された場合に発生する、認識完了イベントを待ち受ける。

▼ 仕様

- type 属性には"speech"、event 属性には"recognize"を指定する。
- match 属性には文法ファイル名と、一つのルール名の組み合わせを記述する。
- 文法ファイル名とルール名の区切りには'#'を使用する（例：match="./grammar.gram#test"）。
- return 属性には認識時の戻り値を格納するための変数を記述する（詳細は付録 1 を参照）。
- 文法は SRGS に従って記述する。
- 発話開始から一定の無音区間までの音声を一つの発話として扱い認識を行う。

▼ エラー

- 音声認識語彙文法が正しく記述されていないとき。

▼ XISL 記述

入力モダリティ	type 属性	操作	event 属性	match 属性	Return 属性
音声	"speech"	認識完了	"recognize"	文法規則のパス	1,2,…:文法規則と同名の変数の集合

▼ 記述例 & 動作例

- XISL 中の<input>記述

```
<input type=" speech " event="recognize" match="./sample.grm#goods" return="goods "/>
```

- 文法ファイル（sample.grm）の記述

```
$goods   =   $book | $cd ;
$book = ぼっちゃん{0001} | こころ{0002} | わがはいはねこである{0003}   ・・・ ;
$cd =  ・・・;
```

- 動作例

  ① ユーザが「ぼっちゃん」と発話

  ② 待ち受け中の<input >イベントが発生（return 属性には 0001 が格納される）

2.1.2 リジェクト（認識失敗）イベント

▼ **機能**

- 指定された音声認識語彙文法にマッチしない発話パタンが入力された場合に発生する、リジェクト(認識失敗)イベントを待ち受ける。

▼ **仕様**

- type 属性は”speech”、event 属性は”reject”を指定する。
- match 属性、return 属性には何も指定しない。

▼ **エラー**

- 特になし

▼ **XISL 記述**

入力モダリティ	Type 属性	操作	event 属性	match 属性	return 属性
音声	“speech”	認識失敗	“reject”	なし	なし

▼ **記述例 & 動作例**

- XISL に書かれている現在待ち受け中の<input>リスト

```
<input type=" speech " event="recognize" match="./sample.grm#goods" return=”goods "/>
<input type=" speech " event="recognize" match="./sample.grm#how_match" />
<input type=" speech " event=" reject " match=" "/>
                                                    ・・・
```

- 文法ファイル（sample.grm）の記述

```
$goods   =   $book | $cd ;
$book =  ぼっちゃん{0001} |  こころ{0002} |  わがはいはねこである{0003}   ・・・ ;
$cd =  ・・・;

$how_match   = $goods   [ は ] いくら;
```

- 動作例

① ユーザが文法ファイルにない語彙を発話（ex.「こんにちは」）

② リジェクトイベントが発生（<input>リストにリジェクトの待ち受けが無い場合は、いずれの<nput>も満たされない）

2.1.3 録音終了イベント

▼ **機能**

- 既に開始されている録音出力の終了イベントを待ち受ける。

▼ **仕様**

- type 属性には”speech”、event 属性には”end_rec”を指定する。
- match 属性には既に開始されている録音出力の録音対象ファイル名を記述する。
- 録音は先に開始させておく必要がある。録音が開始されていない場合、待ち受けているイベントが発生することはない。

▼ **エラー**

- 特になし

▼ **XISL 記述**

入力モダリティ	type 属性	操作	event 属性	match 属性	return 属性
音声	“speech”	認識失敗	“end_rec”	<output> 内の <param name= “name”>要素で指定したファイ名	なし

▼ **記述例 & 動作例**

- XISL 中の<input>記述

```
<input type=" speech " event=" end_rec " match="./test.wav" />
```

- 録音開始<output>の記述

```
<output type=”speech” event=”begin_rec”>
    <![CDATA[
        <param name=”name”>./test.wav</param>
        <param name=”maxtime”>10s</param>
        <param name=”dtmf”>1</param>
        <param name =”finalsilence”>3s</param,>
    ]]>
</output>
```

- 動作例

① 録音開始<output>の出力

② <param>要素に記述されたいずれかの終了条件が満たされる

③ 待ち受け中の<input>イベントの発生

2.2 DTMF

2.2.1 プッシュイベント

▼ 機能

- 指定された文法にマッチするボタンのパタンが入力された場合に発生する、DTMF 入力イベントを待ち受ける。

▼ 仕様

- type 属性には”dtmf”、event 属性には”push”を指定する。
- match 属性には文法ファイル名と、一つのルール名の組み合わせを記述する。
- 文法ファイル名とルール名の区切りには’#’を使用する（例：match=”./dtmf.gram#test”）。
- return 属性には認識時の戻り値を格納するための変数を記述する（詳細は付録 1 を参照）。
- 文法は SRGS に従って記述する。
- DTMF 入力終了時には必ず“#”を入力する必要がある。“#”によって DTMF 入力の末端を判断する。ただし、文法にはこの”#”を記述しない。
- 入力として受け取ることができる DTMF コードは、０～９、*である。#は DTMF 入力の末端を判断するため使用できない。
- 一度に受け取ることができる DTMF コードの個数は、末端の判断のための“#”を除いて 30 個とする。

▼ エラー

- 音声認識語彙文法が正しく記述されていないとき。

▼ XISL 記述

入力モダリティ	type 属性	操作	event 属性	match 属性	return 属性
DTMF	“dtmf”	プッシュ	“push”	文法規則のパス	1,2,…:文法規則と同名の変数の集合

▼ 記述例 & 動作例

- XISL 中の<input>記述

```
<input type=" dtmf " event=" push " match="./dtmf_sample.grm#book" return="book "/>
```

- 文法ファイル（dtmf_sample.grm）の記述

```
$book = 0001 | 0002 | 0003 |  ・・・ ;
```

- 動作例

  ① ユーザが「0001#」と入力

  ② 待ち受け中の<input >イベントが発生（return 属性には 0001 が格納される）

2.3 タイマー入力

2.3.1 タイムアウトイベント

▼ 機能

- 既にカウント中のタイマーのタイムアウト(指定時間を経過時の)イベントを待ち受ける。

▼ 仕様

- type 属性には”timer”、event 属性には”timeout”を指定する。
- match 属性には、待ち受け対象となるカウント中のタイマーID を指定する。
- タイマーは先に生成し、そのタイマーがカウント中である必要がある。

▼ エラー

- 特になし。

▼ XISL 記述

入力モダリティ	Type 属性	操作	event 属性	match 属性	return 属性
タイマー	“timer”	指定時間を経過	“timeout”	<output type=”timer”> で設定した“timer-id”	なし

▼ 記述例 & 動作例

- XISL 中の<input>記述

```
<input type=" timer " event=" timeout " match="sample" />
```

- タイマー生成<output>の記述

```
<output type=” timer” event=” create”>
    <![CDATA[
        <param name=”id”>sample </param>
        <param name=” interval”>60s</param>
        <param name=” start_flag”>TRUE</param>
    ]]>
</output>
```

- 動作例

① 録音開始<output>の出力

② <param>要素に記述されたいずれかの終了条件が満たされる（60 秒経過）

③ 待ち受け中の<input>イベントの発生

3　出力モダリティ

3.1　音声出力

3.1.1　合成音声（TTS）出力モード

▼ **機能**

- テキストを合成音声に変換して出力する。

▼ **仕様**

- type 属性には”speech”、event 属性には”play”を指定する。
- CDATA セクション内で記述する<param>要素の各 name 属性について以下に説明する。
  - ✧ mode(必須)：モード
    合成音声出力の場合は、必ず”TTS”と記述する。
  - ✧ speech-text：読み上げテキスト
    合成音声として出力させたいテキストを記述する。タグを使用することにより部分的に発話タイプを変更できる。読み上げテキストの詳しい説明については下記の「読み上げテキストの記述」を参照されたい。
  - ✧ volume：音量
    トーン音の音量を指定する。音量の種類は以下の 3 種類。
    - silent：小さい
    - normal：通常(デフォルト)
    - loud：大きい
  - ✧ gender：読み上げキャラクター
    テキストを読み上げるキャラクターを指定する。キャラクターの種類は以下の 2 種類。
    - male：男性(デフォルト)
    - female：女性
  - ✧ dtmf：出力終了のための DTMF コード
    音声出力を電話先のユーザが終了させるための DTMF コードを指定する。指定できる DTMF コードは 0～9、*、#のいずれか一つである。デフォルトは#。
- 合成音声出力中のバージインは可能。
- 合成音声出力中には、録音音声出力、トーン出力、録音開始出力を同時に出力できない。
- 必須項目以外の記述において、誤りがある、または値が記述されていない場合はすべてデフォルト値が指定される。

▼ 読み上げテキストの記述

➢ 読み上げタイプの変更

合成音声に変換したい文字列内にタグを記述することにより、部分的に強調して読んだり、発話速度を変化させたりするなどの部分的な読み上げタイプの変更ができる。以下に例を示す。

例 ： おはようございます。<emphasis level = "strong"/> **お元気ですか**。

➢ タグの機能

文字列内で使用できるタグはすべて空タグである。タグを記述した位置以降の文字列に対して指定した読み上げタイプが適用される。一度変更を加えると<reset/>タグなどでデフォルト設定に戻さない限りそのままの読み上げタイプが適用される。そのため、以下の例のように複数の変更を組み合わせることができる。

例 ： おはようございます。<emphasis level = "strong" /> **お元気ですか**。
<speed level="up" /> 今日は良い天気ですね。 <reset /> さようなら。

➢ 使用できるタグ一覧

✧ **break**：指定した時間だけ読み上げを停止する

□ [記述例]：<break time = "1.0s"/>

□ [属性]：time="＊＊＊"

- 読み上げを停止したい時間を指定する。
- 単位は秒で、0 以上で少数第 1 位まで有効。
- 単位として数値の最後に"s"を記述する(省略可能)。

□ 属性を指定しなかった場合は 1 秒（1.0s）間読み上げを停止する。

✧ **emphasis**：読み上げの強調度合いを指定する

□ [記述例]：<emphasis level = "strong"/>文字列

□ [属性]：level="＊＊＊"

- "strong"･･･強調して(音量を上げて)読む
- "moderate"･･･通常の調子で読む(デフォルト)
- "reduced"･･･弱く(音量を下げて)読む

□ 属性を指定しなかった場合はデフォルトを設定する

✧ **say-as**：英字、数字の読み上げ方を指定する

□ [記述例]：<say-as type = "spell-out"/>文字列

□ [属性]：type="＊＊＊"

- "spell-out"･･･英文字を一文ずつ読み上げる
- "cardinal"･･･数字を一文字ずつ読み上げる
- "normal"･･･英字は通常通り読み、数字は桁読みする(デフォルト)

- □ 属性を指定しなかった場合はデフォルトを設定する
- ✧ **speed**：読み上げる発話速度を指定する
  - □ [記述例]：<speed level = "up"/>文字列
  - □ [属性]：level=”＊＊＊”
    - “up”･･･発話速度を上げて読み上げる
    - “default”･･･通常の発話速度で読み上げる(デフォルト)
    - “down”･･･発話速度を下げて読み上げる
  - □ 属性を指定しなかった場合はデフォルトを設定する
- ✧ **reset**：読み上げタイプをすべてデフォルト設定に戻す
  - □ [記述例]：<reset/>文字列
  - □ [属性]：なし

➢ **注意事項**

- ✧ 合成音声生成時、タグの前後でテキストは分割され、別々に処理される。このため生成する合成音声には、タグの前後の文法的な接続などが考慮されない。
- ✧ “<”と”/>”はタグの一部として処理されるので、タグ以外の目的で使用できない。
- ✧ 句点、読点は読み上げられない。
- ✧ 空白とタブは句点、改行は読点と同様に扱われる。
- ✧ 文字列の最初の句点､読点は無視される｡タグ直後の句点､読点も同様に無視される。
- ✧ “／”、”・”、”：”で区切られた数字を日付や時刻として読み上げる。
- ✧ 数字の前にある単位記号を､数字の後に入れ替えて読む｡単位入れ替えができる記号は“¥”、“$”、“￠”、“￡”の４つである。（例：「¥200」＝「にひゃくえん」）
- ✧ 「 」、{ }、( )などの括弧は読み上げられない。
- ✧ システムの処理速度によっては､長い文を合成音声に変換する場合若干時間がかかる。
- ✧ 一つでもエラーがあると合成音声に変換されない。
- ✧ ここでの記述方は､東芝音声合成コントロールの読み上げテキスト記述方に従っている。詳細についてはそのヘルプを参照のこと。

▼ **エラー**

- ➢ 読み上げテキストに誤りがあるとき。
- ➢ name 属性の必須項目の記述に誤りがある、または記述されていないとき。

▼ XISL 記述

出力モダリティ	type 属性	操作	event 属性	CDATA セクション内で記述する<param>要素	
				<param>の name 属性	<param>の内容
音声	“speech”	再生	“play”	“mode” （必須）	“TTS”
				“speech-text”	読み上げテキスト
				Volume	音量 “silent”, ”normal”, ”loud” デフォルト：”normal”
				“gender”	“male” or “female” デフォルト：”male”
				“dtmf”	出力を終了するための DTMF コード（0～9、#、*） デフォルト：#

- <param name=”speech-text”>内で記述できる要素

要素名	属性名	属性値	機能
<break/>	time	時間 (ex.“1s”) デフォルト：1.0s	属性 time に指定した期間、出力を停止する
<emphasis/>	level	“strong”, ”moderate”, ”reduced” デフォルト：”moderate	タグ以降のテキストを強調して出力する。強調度は属性 level で指定する
<say-as/>	type	“spell-out”	タグ以降の英文字を一文字ずつ読み上げる
		“cardinal”	タグ以降の数字を一文字ずつ読み上げる
		“normal”（デフォルト）	英字は通常通り読み、数字は桁読みする
<speed/>	level	“up”, “default”, “down” デフォルト：”default”	タグ以降の発話速度を変更する。速度レベルは 3 段階
<reset/>	なし	なし	読み上げマークアップを全てリセットする

▼ 記述例

- TTS 再生<output>の記述

```
<output type="speech" event="play">
    <![CDATA[
        <param name=”mode”>TTS</param>
        <param name=" speech-text ">
            入力された ID <speed type=”down”/><emphasis level=”strong”/>
          <say-as type=”cardinal”/>12345 は、間違っています。
          <break time=”1s”/>もう一度入力してください。
        </param>
    ]]>
</output>
```

3.1.2 録音音声出力モード

▼ **機能**

- 指定された WAVE ファイルを再生出力する。

▼ **仕様**

- type 属性には”speech”、event 属性には”play”を指定する。
- CDATA セクション内で記述する<param>要素の各 name 属性について以下に説明する。
  - ✧ mode(必須)：モード
    録音音声出力の場合は、必ず”rec_speech”と記述する。
  - ✧ target：ファイル名
    再生したい WAVE ファイルのファイル名を指定する。URI 記述も可能。ただし、本バージョンのフロントエンドでは、WAVE ファイルを電話サーバ上に保存しておく。
  - ✧ volume：音量
    トーン音の音量を指定する。タイプは以下の 3 段階がある。
    - silent：小さい
    - normal：通常(デフォルト)
    - loud：大きい
  - ✧ dtmf：出力終了のための DTMF コード
    録音音声出力を電話先のユーザが終了させるための DTMF コードを指定する。指定できる DTMF コードは 0～9、*、#のいずれか一つである。デフォルトは#。
- 録音音声出力が可能なファイル形式は WAVE ファイルのみ。さらに、WAVE ファイルの再生可能なフォーマットは以下のいずれかである。
  - 8 または 11KHz linear 8 ビット
  - 8 または 11KHz mu-law 8 ビット
  - 8 または 11KHz a-law 8 ビット
- 録音音声出力中のバージインは可能。
- 録音音声出力中には、合成音声出力、トーン出力、録音開始出力を同時に出力できない。
- 必須項目以外の記述において、誤りがある、または値が記述されていない場合はすべてデフォルト値が指定される。

▼ **エラー**

- 指定した再生ファイルが開けない場合。
- 指定した再生ファイルのファイル形式またはフォーマット形式がサポートされていない場合。
- name 属性の必須項目の記述に誤りがある、または記述されていない場合。

▼ XISL 記述

出力モダリティ	type 属性	操作	event 属性	CDATA セクション内で記述する<param>要素	
				<param>の name 属性	<param>の内容
音声	speech	再生	play	“mode”（必須）	“rec_speech”
				“target”（必須）	ファイルの URI
				“volume”	音量 “silent”, ”normal”, ”loud” デフォルト：”normal”
				“dtmf”	出力を終了するための DTMF コード（0～9、#、*） デフォルト：#

▼ 記述例

- 録音音声ファイル再生<output>の記述

```
<output type="speech" event="play">
    <![CDATA[
        <param name=”mode”>rec_speech</param>
        <param name=" target ">./test.wav</param>
        <param name=" volume ">silent</param>
        <param name=" dtmf ">#</param>
    ]]>
</output>
```

3.1.3 録音開始出力

▼ **機能**

- 録音を開始する。
- 録音終了イベントについては、『2.1.3 録音終了イベント』を参照されたい。

▼ **仕様**

- type 属性には”speech”、event 属性には”begin_rec”を指定する。
- CDATA セクション内で記述する<param>要素の各 name 属性について以下に説明する。
  - ✧ name(必須)：保存先ファイル名

    録音した音声データを保存したいファイルの名前を記述する。URI 記述も可能。ただし、本バージョンのフロントエンドでは、WAVE ファイルを電話サーバ上に保存する。
  - ✧ maxtime：録音時間

    録音可能な時間（最大時間）を記述する。単位は秒で、0 以上、少数第 1 位まで有効である。単位として数値の最後に”s”を記述する(省略可能)。デフォルトは 60 秒。最大は 60 秒。（記述例：”12.0s”, “20”）
  - ✧ dtmf：録音終了のための DTMF コード

    録音を電話先のユーザが終了させるための DTMF コードを指定する。指定できる DTMF コードは 0～9、*、#のいずれか一つである。デフォルトは#。
  - ✧ finalsilence：録音終了のための無音時間

    録音を終了させるための終了条件となる無音時間を記述する。ここに記述した時間だけ無音が続くと録音を終了する。単位は秒で、0 以上、少数第 1 位まで有効である。単位として数値の最後に”s”を記述する(省略可能)。デフォルトは 60 秒。最大は 60 秒。（記述例：”12.0s”, “20”）
- 音声データは以下のファイル形式で録音される。
  - 11KHz linear 8 ビット
- 録音中は、合成音声出力、録音音声出力、トーン出力を同時に出力できない。
- 作成されたファイルは電話サーバ実行フォルダー上に保存される。
- 必須項目以外の記述において、誤りがある、または値が記述されていない場合はすべてデフォルト値が指定される。

▼ **エラー**

- 録音音声データの保存用ファイルが開けないとき。
- name 属性の必須項目の記述に誤りがある、または記述されていないとき。

▼ XISL 記述

出力モダリティ	type 属性	操作	event 属性	CDATA セクション内で記述する<param>要素	
				<param>の name 属性	<param>の内容
音声	“speech”	録音開始	“begin_rec”	“name”（必須）	録音対象のファイル名
				“maxtime”	最大録音時間 (秒：少数第一位まで) デフォルト：60s
				“dtmf”	出力を終了するための DTMF コード（0～9、#、*） デフォルト：#
				“finalsilence”	録音の終了条件 となる無音時間の長さ デフォルト：60s

※　終了条件が未指定の場合　⇒　『maxtime=60s, dtmf=#, finalsilence=60s』　となる

▼ 記述例 & 動作例

- 録音開始<output>の記述

```
<output type=”speech” event=”begin_rec”>
    <![CDATA[
        <param name=”name”>./test.wav</param>
        <param name=”maxtime”>10s</param>
        <param name=”dtmf”>1</param>
        <param name =”finalsilence”>3s</param,>
    ]]>
</output>
```

- XISL 中の<input>記述

```
<input type=" speech " event=" end_rec " match="./test.wav" />
```

- 動作例

① 録音開始<output>の出力

② 終了条件が満たされる（10s 経過 or DTMF’1’がプッシュ or 無音区間を 3s 間検知）

③ 待ち受け中の<input>イベントの発生

3.2 トーン出力

▼ 機能

- トーン音を出力する（音の種類は low、normal、high の 3 種類）。

▼ 仕様

- type 属性には”tone”、event 属性には”play”を指定する。
- CDATA セクション内で記述する<param>要素の各 name 属性について以下に説明する。
  - ✧ duration：出力時間

    トーン音の出力時間を指定する。単位は秒で、0 以上、少数第 1 位まで有効である。単位として数値の最後に”s”を記述する(省略可能)。デフォルトは 1 秒。最大は 60 秒。（記述例：”1.2s”, “2”）
  - ✧ type：音の種類

    トーン音の種類には以下の 3 種類がある。
    - low：低音
    - normal：通常の音(デフォルト)
    - high：高音
  - ✧ volume：音量

    トーン音の音量には以下の 3 段階がある。
    - silent：小さい
    - normal：通常(デフォルト)
    - loud：大きい
  - ✧ dtmf：出力終了のための DTMF コード

    トーン出力を電話先のユーザが終了させるための DTMF コードを指定する。指定できる DTMF コードは 0～9、*、#のいずれか一つである。デフォルトは#。
- トーン出力中のバージインはできない。
- トーン出力中には、合成音声出力、録音音声出力、録音開始出力を同時に出力できない。
- 必須項目以外の記述において、誤りがある、または値が記述されていない場合はすべてデフォルト値が指定される。

▼ XISL 記述

出力モダリティ	type 属性	操作	event 属性	CDATA セクション内で記述する<param>要素	
				<param>のname属性	<param>の内容
トーン	“tone”	トーン出力	“play”	“duration”	出力する期間 (秒：少数第一位まで) デフォルト：1.0s
				“type”	トーンのタイプ “low”, “normal”, “high” デフォルト：”normal”
				volume	音量 “silent”, ”normal”, ”loud” デフォルト：”normal”
				“dtmf”	出力を終了するための DTMF コード（0～9、#、*） デフォルト：#

▼ 記述例

- 録音音声ファイル再生<output>の記述

```
<output type="speech" event="play">
    <![CDATA[
        <param name=”mode”>rec_speech</param>
        <param name=" target ">./test.wav</param>
        <param name=" volume ">silent</param>
        <param name=" dtmf ">#</param>
    ]]>
</output>
```

3.3 タイマー出力

▼ 機能

- タイマーの生成、開始、停止、一時停止、削除、再設定を行う。
- タイマー入力の受け取りについては、『2.3.1 タイムアウトイベント』を参照されたい。

▼ 仕様

- type 属性には”timer”を指定する。
- event 属性に指定する各属性値ついて以下に説明する。
  - ✧ create：生成

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：生成するタイマーの ID
    - duration：タイマーの間隔〔秒〕（デフォルトは 60s）
    - start_flag：タイマー生成時にカウントを始めるかどうか
      - ✓ ”TRUE”→生成時にカウント状態にする(デフォルト)
      - ✓ “FALSE”→生成時に停止状態にする
  - ✧ start：開始

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：開始するタイマーの ID
  - ✧ stop：停止

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：停止するタイマーの ID
  - ✧ pause：一時停止

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：一時停止するタイマーの ID
  - ✧ delete：削除

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：削除するタイマーの ID
  - ✧ reset：間隔の再設定

    CDATA セクション内で記述する<param>要素の各 name 属性について以下に示す。
    - id(必須)：再設定するタイマーの ID
    - duration：タイマーの間隔〔秒〕（デフォルトは 60s）
- 生成したタイマーの ID は、一つのセッションにおいて常に有効である。ただし、タイマー削除後は無効となる。
- タイマーの ID には任意の文字列を指定可能である。最大文字数は半角で 128 字、全角で 64 字とする。タイマーID は必須項目であるので必ず記述する。
- タイマー生成時、start_flag が TRUE のときはすぐにカウント状態に、FALSE のときは停止状態にする。start_flag を指定しないときはすぐにカウント状態にする。
- duration の値は秒間隔で指定し少数第一位まで有効。数字の後に単位”s”を付ける(省略可能)。最小値は 0 秒、最大値は 3600.0 秒とする。省略すると 60 秒となる。（記述

例：”5.5s”，”3”）

- タイマーはタイムアウト後、停止状態となる。
- カウント状態のタイマーは明示的に stop、pause、delete しない限り、セッション中は動きつづける。属性 duration で指定した時間に達した場合は、タイムアウトし停止状態となる。
- システムが許す限りタイマーはいくつでも生成可能である。
- タイマーは厳密なものでなく、システムの状態によっては実時間より若干のタイムラグが起こる可能性がある。
- 必須項目以外の記述において、誤りがある、または値が記述されていない場合はすべてデフォルト値が指定される。

▼ 動作

- **タイマーの状態**
  - ✧ カウント状態：タイマーがカウントをしている状態。経過時間が duration（タイムアウトまでの間隔）まで達したときタイムアウトする。
  - ✧ 停止状態：タイマーが停止している状態。経過時間は 0。
  - ✧ 一時停止状態：タイマーが一時停止している状態。経過時間はタイマー開始からの時間。
- **各状態における動作**
  - ✧ カウント状態のとき
    - start：無視する。
    - stop：カウントを停止する。経過時間を 0 に戻す。停止状態になる。
    - pause：カウントを一時停止する。経過時間はそのまま。一時停止状態になる。
    - delete：タイマーを削除する。
    - reset：カウントの間隔を再設定する。経過時間はそのまま。カウント状態は維持される。経過時間よりカウント間隔を少なく設定すると、すぐにタイムアウトする。
  - ✧ 停止状態のとき
    - start：0 からカウントを始める。カウント状態になる。
    - stop：無視する。
    - pause：無視する。
    - delete：タイマーを削除する。
    - reset：カウントの間隔を再設定する。一時停止状態は維持される。
  - ✧ 一時停止状態のとき
    - start：経過した時間からカウントを始める。カウント状態になる。
    - stop：経過時間を 0 に戻す。停止状態になる。
    - pause：無視する。
    - delete：タイマーを削除する。
    - reset：カウントの間隔を再設定する。経過時間はそのまま。一時停止状態は維持される。経過時間よりカウント間隔を少なく設定すると、カウント開始後すぐ

にタイムアウトする

▼ **エラー**

- すでに宣言した ID を指定してタイマーを生成しようとしたとき。
- 存在しない ID を指定し、タイマーを開始、停止、一時停止、削除、再設定しようとしたとき。
- タイマーID の文字数が最大値（半角 128 文字、全角 64 文字）を超えているとき。
- duration の値が正の数以外のとき。または最大値（3600.0 秒）を超えているとき。
- name 属性の必須項目の記述に誤りがある、または記述されていないとき。

▼ **XISL 記述**

出力モダリティ	type 属性	操作	event 属性	CDATA セクション内で記述する<param>要素	
				name 属性	<param>の内容
タイマー	“timer”	生成	“create”	“id”（必須）	生成するタイマーの ID
				“duration”	イベントを発生させる間隔 (秒：少数第一位まで) (ex. “60s”) デフォルト：60s
				“start_flag”	同時にカウントを開始するかどうか[省略可] "TRUE"→開始（デフォルト） "FALSE"→停止したまま
		開始	“start”	“id”（必須）	操作対象タイマーの ID
		停止	“stop”		
		一時停止	“pause”		
		削除	“delete”		
		間隔の再設定	“reset”	“id”（必須）	操作対象タイマーの ID
				“duration”	イベントを発生させる間隔 デフォルト：60s

▼ **記述例**

- 2.3.1 タイムアウトイベントの記述例 & 動作例を参照

付録 1：音声（DTMF）文法記述の仕様

1. はじめに

この文書は PC 用フロントエンドと電話用フロントエンドの音声入力で使用される認識文法を規定する。

文法の記述には SRGS(Speech Recognition Grammar Specification)を使用する。

2. 記法

2.1 トークン（Tokens）

空白、あるいは特別な構文機能を持つ記号（ ; = | () [] {} ）によって区切られた単位の事をいう。

```
hello
bon voyage
this is a test        // ４つのトークンからなる順列
“San Francisco”
2
```

なお、引用符で囲まれたトークン内には空白があってもよい。以下は全て同等なトークンとして扱われる。

```
“San Francisco”
“San
Francisco”
“      San          Francisco“
```

2.2 ルール参照（Rule References）

今回はローカル参照のみ使用する。ローカルルール参照では、参照先が同じグラマー内に存在しており、常にローカルなルール名だけから成る単純なルール名参照を用いる。
ルール参照は、ルール名の頭に$をつけることで行う。

```
$city
$digit
```

2.3 順列（Sequences）

トークン、ルール参照が一続きとなった順番列。

```
this is a test                    // トークンの順列
$action $object    // ルール参照の順列
the $object is $color          // トークンとルール参照の順列
```

集合は”()”によって区切られても良い。

```
Adachi | Kobayashi | Nakamura | $otherNames
( 1 | 2 | 3 )
```

2.4 タグ（Tags）

タグ”{}”内の記述は、前に記述されているトークンやルール参照に関連付けられる。発話内容に替わって、タグ内の値が<input>に渡されるようになる。

```
hamburger   |   burger {hamburger}   |   ( chicken [sandwich] ) {chicken}
```

2.5 その他

(...)	グルーピング
[...]	省略可能

“()”で囲われたトークンやルール参照はグループ化される。

ルール中に“[]”で囲われたトークンやルール参照があれば、それがない発話でもそのルールに適合させることができる。

3. ルール定義

ルール定義は以下のように記述される。

```
// ABNF 形式
$ruleName = ruleExpansion;
```

```
// 例
$city =  とよはし | さやま | かなざわ
$exp = $city [の] へるぷ;
```

4. XISL の記述

<input>の match 属性にパスを含むファイル名と参照するルール名を指定する。ファイル名とルール名は”#”で区切ること。return 属性には複数の変数を記述できる。変数には認識した語彙か、もしあるれば 2.5 のタグに記述した値が格納される。
以下に<input>および、グラマーファイルの記述例を示す。

```
<input type=”speech” event=”recognize” match=”grammar.txt#purchase” return =”goods” />

grammar.txt
$goods = てぃーしゃつ | ぱんつ | そっくす | くつした{そっくす};
$num = ( いっこ| ひとつ){1} | (にこ | ふたつ ){2} | ( さんこ | みっつ ){3};
$bug = こうにゅう | かう | ください;
$purchase = $goods [を] $num [$buy];
```