2005－SLP－58（3）
2005／10／21

話題変更入力の取り扱いに着目した対話方式

屋野 武秀，阿部 一彦，上原 龍也，正井 康之

（株）東芝 研究開発センター マルチメディアラボラトリー
〒212-8582 神奈川県川崎市小向東芝町 1
takehide.yano@toshiba.co.jp

複数の話題を取り扱う対話において，ユーザが対話対象の話題を変更する話題変更入力を取り扱い，対話の柔軟性を向上させる対話方式を提案する．提案方式は全話題間の遷移手順が不要な方式であり，話題別の対話シナリオを参照して対話を進行する．また，対話実行中に発生するユーザ入力時の直接的な話題変更，話題終了時に判明する間接的な話題変更に対応するため動的に話題選択を行う．提案方式をカーナビの目的地設定タスクに適用した対話システムで対話実験を行った．実験中の対話ではユーザ発話の 56%が直接的に話題変更を示唆し，また話題終了時の 23%の事例に間接的な話題変更による影響があった．そのような状況下で誤動作要因となった話題選択処理は全体の 0.9%であり，提案方式が対話の柔軟性向上に寄与することを示した．

A Dialogue Management with Handling Changing Topic

Takehide Yano, Kazuhiko Abe, Tatsuya Uehara, Yasuyuki Masai
Multimedia Lab., Corporate Research & Development Center, Toshiba Corp.
1, Komukai-Toshiba-Cho, Saiwai-ku, Kawasaki, 212-8582, Japan

In this paper, we propose a dialogue management (DM) method with handling changing topic by user's input, for flexibility of DM. The dialogue scenario of proposed method is only the scenario of each topic, and there is no need to design transitions between topics. Proposed method selects automatically the topic that DM should talk about, when user input, and when dialogue of a topic is ended. Simulation results show that the changing topic occurred in 56% of the user's input and 23% of the end of the dialogue of a topic, and the error rate of topic selection is 0.9%. The result also shows that proposed method contributes for flexibility of the DM.

1. はじめに

音声対話インタフェースを操作が複雑な実タスクに適用するためには複数の話題を扱う対話方式が必要である．例えば，音声インタフェースの導入が進んでいるカーナビゲーションシステム(カーナビ)は，場所検索(住所やジャンルなど)，目的地・経由地設定，詳細情報の問い合わせといった複数の話題(コマンド)を扱う．

複数の話題を扱う対話システムがユーザ主導型の発話を許容すると，ユーザは任意の話題を指定するため，話題変更の意図を持つユーザ入力である**話題変更入力**を考慮せねばならない．また，システムが扱う話題には『目的地設定は場所検索の結果を利用する』といった依存関係があるものも存在するため，システムは依存関係を考慮して複数の話題を遷移する対話を行う必要がある．話題変更入力を取り扱わない対話システムとの対話においては，ユーザはシステムが決定した話題間の遷移順序に従って対話せねばならず，対話進行の大きな制約となる．対話の柔軟性を向上させるために話題変更入力を取り扱う必要がある．

話題変更入力を取り扱うために，対話シナリオ設計者が話題変更入力のパタンを予測し，予め対話シナリオに動作を記述することが考えられる．しかし話題変更入力は対話方式が扱う話題の組合せに応じて増加する．単純なタスク[1][2][3]であれば組合せも検討可能であるが，カーナビのような複雑なタスクになると話題の組合せは膨大になるため，話題変更入力の予測は困難である．

話題変更入力に対する動作を動的に決定する方式として，話題変更入力によって導入された話題を対話に割り込ませ，その話題の対話が終了した後に元の対話状態に戻る手法[4]が提案されている．しかし話題変更入力により元の話題の再開が不要となる場合があるため，話題変更入力による話題を割り込ませるだけでは不十分である．

例えば，図１の対話例では話題「目的地設定」の対話で目的地を決定するためにシステムは話題「住所検索」の対話を開始している(S1，S3)．しかし対話例(a)では直接的に(U2)，対話例(b)は話題「ジャンル検索」を割り込ませて(U4～S5)間接的にユーザは目的地候補となる場所“レストランＡ”

を指定し，「目的地設定」の対話を選択している．これに伴い目的地の場所を決定するために利用していた「住所検索」の対話は不要となる．しかし従来のように元の話題に戻るだけでは，いずれの場合も「住所検索」の対話を再開してしまう．その結果ユーザが場所を指定しているにも関わらず S2,S6 を出力せずに「都道府県名をどうぞ」という応答をユーザに提示してしまう．

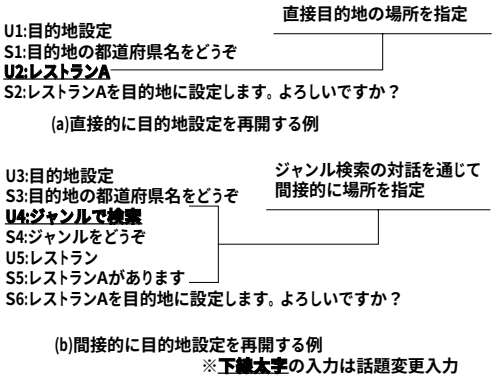

図 1 話題選択を行う対話例

以上のように，ユーザ入力や対話結果によって対話対象の話題が変更されたり，ある話題の対話が不要となったりする場合がある．従って，話題変更入力を動的に取り扱う対話方式には対話を進行させる話題を選択すると共に対話状態を更新する**話題選択処理**が必須である．

本稿では話題変更入力を容易に取り扱う対話方式を提案する．提案方式は[8]を拡張した方式で，全話題間の遷移手順が不要な対話方式であり，話題別の対話シナリオを参照して対話を進行する．また，直接的・間接的な話題選択を動的に行うためにユーザ入力時と各話題の対話が終了した時に動的に話題選択処理を行う．

以下，2 章では提案方式の説明を行い，3 章では提案方式を利用した対話システムについて述べる．4 章で対話システムを用いた実験について述べ，考察を加える．

2. 提案方式

2.1. 話題スタックを用いた対話管理（基本方式）

提案方式では対話によってタスクを遂行する対話行為を対象とする．各タスクを個々の話題と対応付け，各話題についてその話題の対話を実行するための対話手順を記述した対話シナリオと，対話終了のための必須属性等を指定するフレーム定義とを対にした情報を予めデータベースに与えておく．これらの情報は対話進行中に適宜参照される．提案方式では全話題の遷移を指定する対話シナリオは必要としない．

提案方式は単一のユーザ発話の解釈結果を入力とする．ユーザ発話の解釈結果はその発話に対応する話題に関するフレームか任意の値の形式(図 2)に変換されるものとする．

発話	入力（発話解釈結果）
『○○に目的地設定』	目的地設定（目的地：○○）
『ジャンルで検索』	ジャンル検索（指定ジャンル：φ）
『レストランA』	"レストランA"<場所型>

フレームの書式は『話題名(属性名：属性値、…)』。
値の書式は『"値"<型>』。
"φ"は属性値が代入されていないことを示す。

図 2 基本方式に入力される情報の例

提案方式は話題情報のスタック（話題スタック）を用いて複数の話題を用いた対話制御を行う(図 3)．話題情報は各話題に関して具現化したフレームと対話の進行状況を対にした情報である．フレーム形式の情報から属性値代入状況に応じた対話進行状況の話題情報を作成できる．この話題スタックを利用した基本的な対話方式（以下基本方式と呼ぶ）について説明する．

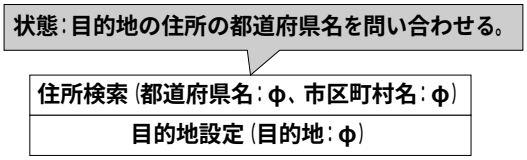

図 3 話題スタック

基本方式は常に話題スタックのトップにある話題情報を対話対象(**TT**)としてユーザとの対話を行う．各話題情報のフレームにある必須属性の取得をサブゴールとする．**TT** の対話は **TT** の話題の対話シナリオを参照して未解決なサブゴールを解決するように進行し，全サブゴールを解決すると終了する．

TT のサブゴールを解決するための対話手続きとして別の話題を利用する場合にはその話題のフレームが導入される．基本方式はそのフレームに対応する話題情報を作成し，新たな **TT** として話題スタックにプッシュする．そして新たな **TT** の対話を開始し，元の **TT** 対話は一時停止される．

図 3 は目的地を決定するために話題「住所検索」の話題情報をプッシュし，対話状態が「目的地を住所で決定」に更新された話題スタックである．その後「住所検索」を **TT** として対話を開始する．

TT の対話が終了すると，基本方式は **TT** をポップし，その後話題スタックのトップになった話題

情報を **TT** として対話を再開する．対話を進行すべき話題情報は **TT** であり，それ以外の話題情報は対話の再開を待つ話題情報である．対話は話題スタックが空になるまで継続される．

2.2. 提案方式

提案方式は基本方式に話題変更入力を取り扱うための話題選択処理を加えた方式である．図 1 の対話例で示したように話題選択処理はユーザ入力時と話題の対話終了時に行う必要がある．これらの話題選択処理を取り扱うために提案方式は図 4 のように動作する．

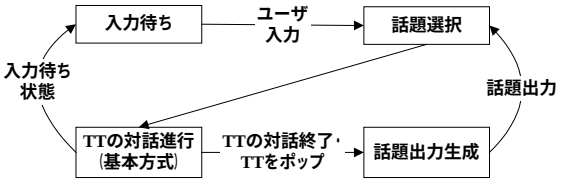

図 4 提案方式のアルゴリズム

提案方式はユーザ入力時に話題選択処理を実行する. 話題選択処理はユーザ入力に基づいて**TT**を選択し，話題スタックを更新する処理である．更新の際にはユーザ入力を話題スタック内に反映させ，更新後の話題スタックを対象としてユーザとの対話を継続する．

図 1(b)の対話例のように，間接的な話題変更はある話題の対話結果を利用して話題選択を行うことに相当する．提案方式ではこれを扱うために**話題出力**を利用した話題選択処理を行う．話題出力はユーザが **TT** の対話結果として得る情報である．この話題選択処理の終了後も更新後の話題スタックを対象として対話を継続する．

提案方式は話題出力を生成するために，各話題の対話シナリオに話題出力の定義を追加する．話題出力は各話題をある程度抽象化した情報を指定する．例えば話題「住所検索」「施設検索」のいずれも話題出力は検索結果の場所と定義できる．話題出力生成処理ではこの定義を参照して話題出力を生成する．

2.3. 話題選択処理

2.3.1. 概要

話題選択処理は **TT** を含む話題スタック内の話題情報を入力で更新して対話再開(**Restart**), 入力で指定されたフレームをユーザが割り込ませた新たな **TT** としてプッシュ(**Push**), 入力を無視して同じ動作を繰り返す(**Ignore**)の中から動作を決定し, 決定内容に基づき話題スタック更新を行う．

TT の対話を継続する入力の場合は **TT** を **Restart** するように動作が決定される．話題選択処理の動作フローチャートを図 5 に示す.

話題選択処理では **Restart** か否かの判定を行う．判定結果が **Restart** でない場合は入力がフレームであれば話題情報が作成可能であるので **Push** とし，それ以外は **Ignore** と判定する．また判定結果に応じて対話再開が不要な話題情報(不要話題)を消去する話題スタック更新処理((a),(b))を行う．

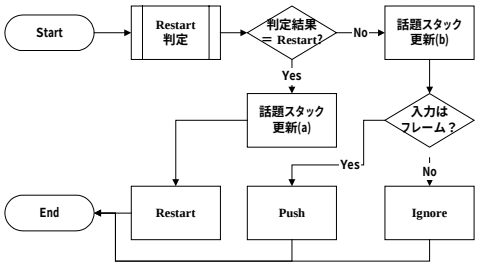

図 5 話題選択処理

2.3.2. Restart 判定

Restart判定ではユーザに対話再開を示唆された話題情報を話題スタック内から特定し，特定できない場合は**Restart**ではないと判定する. 各話題情報は関連する話題の対話進行状況を管理しており，ある話題の対話を進行させるために，ユーザはその話題の対話進行状況の更新に寄与する内容を入力する. 従って**Restart**する話題情報は入力を受け取ってその内容を更新する**Catch処理**(図 6)が可能な話題情報とすることができる．話題スタック内に入力を**Catch**することが可能な話題情報が複数存在する場合は，**TT**に近いものを**Restart**する話題情報として選択する．各話題情報が**Catch**できる情報は同種のフレーム（話題情報のフレームを上書き）と，各話題情報の**Catch**条件(図 6 左)に該当するものがある．

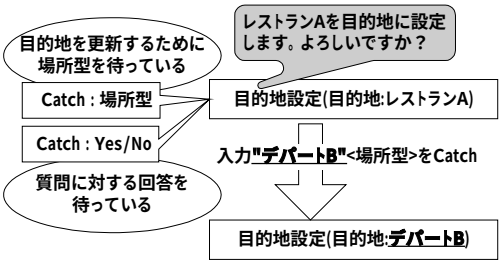

図 6 Catch 処理

提案方式では各話題情報における **Restart** 可能な情報を指定する条件として各話題のシナリオに **Catch** 条件を付与する(図 6 左). **Catch** 条件には例えばその時に解決対象となっているサブゴール

や，システムプロンプトに対するユーザ応答などの情報がある．**Catch** 条件は各話題情報の対話進行状況によって異なるため，提案方式はユーザとの対話進行状況に応じて対話シナリオから必要な **Catch** 条件を取得して話題情報に格納する．

2.3.3. 話題スタック更新

判定結果が **Restart** の場合(a)は，選択された話題情報より後にプッシュされた話題情報を不要話題として消去する．提案方式ではサブゴールを解決する手段，或いはユーザが割り込ませた話題として新たな話題情報を話題スタックにプッシュするので，**Restart** する話題のサブゴールの解決や割り込ませた話題の対話が不要と考えられる．消去後は **Restart** 判定で選択された話題情報が話題スタックのトップ(**TT**)になる．

判定結果が **Restart** ではない場合(b)は，入力と同等の効果をもたらす話題情報を不要話題として消去する．例えば，場所検索中に別の場所が入力された時には，場所検索の対話が不要であると考えられる．提案方式では各話題の対話シナリオにある話題出力定義と入力が同種の情報であれば同等の話題情報と判定し，話題スタックのトップから順に判定して同等でないものが現れるまで話題情報を消去する．

2.4. 動作例

図 1(b)の対話例に対する提案方式の動作説明を図 7 に，U5 に対する話題出力生成・話題選択処理を図 8 に示す．提案方式はユーザ入力・話題出力と **Catch** 条件を関連付けて対話管理を行う．この仕組みにより直接的・間接的な話題変更指定に対しても提案方式は再開すべき話題を選択することが可能となる．

2.5. 考察

2.5.1. 従来方式との比較

従来方式として話題をスタックで管理しユーザ入力時に話題選択を行う方式[5]が提案されている．しかし，この方式には話題出力がなく，間接的な話題変更には対応していない．提案方式は入力と話題出力に基づき話題選択を行うので，間接的な話題変更にも対応している．

対話結果(話題出力に相当)を話題選択に利用する方式として，話題を順不同に管理し，他の話題から提供される情報を取得する方式[6]や，話題間の依存関係に基づく木構造で表現した話題構造を利用して対話の結果を親の話題が取得する方式[7]が提案されている．しかし，例えば複数の話題が依存する話題(X)が存在すると，親の話題は一つではなく，また X の対話が終了した時には情報を取得する話題を一意に特定できない場合がある．提案方式は対話に現れた話題情報の順序を表現する話題スタックを利用しており，情報を **Catch** する話題情報を選択する際には順序情報を用いて一意に決定することが可能である．

対話例	動作説明
U3:目的地設定	入力解釈結果＝目的地設定（目的地：φ） 初期段階はCatch可能な話題情報がないのでPush判定 目的地設定の対話シナリオにより場所決定方法として [住所検索]をプッシュ
S3:目的の都道府県名をどうぞ	Catch条件は[目的地設定]が場所型の値、[住所検索]が 都道府県名の値
U4:ジャンルで検索	入力解釈結果＝ジャンル検索（指定ジャンル：φ） 入力はCatch条件に該当しないフレームなのでPush判定
S4:ジャンルをどうぞ	[ジャンル検索]のCatch条件はジャンル型の値
U5:レストラン	入力解釈結果＝"レストラン"<ジャンル> 入力は[ジャンル検索]のCatch条件に該当するので、 [ジャンル検索]のフレームを次に更新してRestart ジャンル検索（指定ジャンル：レストラン） [ジャンル検索]はジャンル情報を受理して対話終了 周辺のレストランを検索し結果として"レストランA"を得る。 ジャンル検索をポップ（図8①）
S5:レストランAがあります	話題出力＝"レストランA"<場所型>（図8②） 話題出力は[目的地設定]のCatch条件に該当する(図8③) ので、[目的地設定]のフレームを次に更新してRestart 目的地設定（目的地：レストランA） [住所検索]は不要話題として話題スタックから削除（図8④）
S6:レストランAを目的地に設定します。よろしいですか？	[目的地設定]のCatch条件にYes/Noを追加（図8④）

※ "[話題名]"はその話題の話題情報を指す

図 7 提案方式の動作例

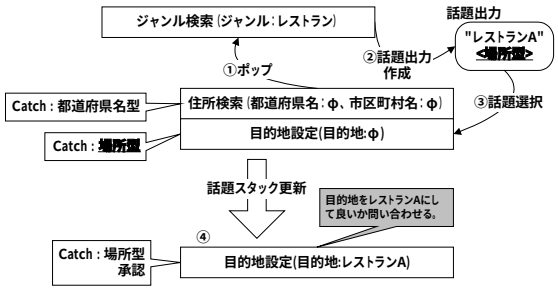

図 8 間接的な話題選択の処理例

2.5.2. シナリオ設計コスト

提案方式は動的な話題選択を行うので対話シナリオに全話題間の遷移を指定する必要がない．一方，提案方式では話題選択処理のために **Catch** 条件と話題出力を各話題の対話シナリオに付与する必要があるが，**Catch** 条件は話題内の進行状況のみに依存し，話題出力は対象の話題の種類のみに依存する条件である．従って提案方式は対話における局所的な状態のみを考慮すれば話題選択処理を実行することが可能であり，対話シナリオ設計コストの抑制に寄与するものと考えられる．

3. 対話システム

提案方式を利用した対話システムを開発した．対話システムは入力解釈部，提案方式を利用した対話管理部，対話シナリオを管理する話題 DB で構成される(図 9). 対話管理部はユーザとの対話実

行中にアプリケーションにコマンドの実行を依頼し, その結果を受理してその後の対話に利用する.

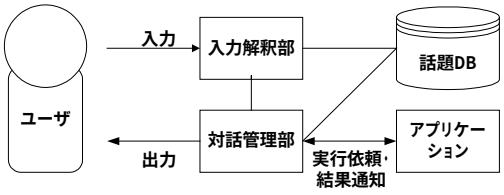

図 9 対話システムの構成図

```
1   // フレーム定義
2   SetDest(Place型:Dest);   // 目的地設定のフレーム(Place型引数)
3    Place Address(Pref型:SetPref,City型:SetCity);
4                            // 住所検索のフレーム(Place型話題出力)
5
6   // シナリオ定義
7   Scenario SetDest {       // SetDest(Dest(Place型))の対話シナリオ
8     Int nOKFlag = FALSE;   // 承認フラグ
9     State ConfirmDest();   // 問い合わせ処理 (部分話題情報)
10    SubGoal : {            // SubGoal指定
11      case !IsValidDest() :  LoadScenario(Address(φ,φ));
12                  // Destが代入されていないなければ住所検索をプッシュ
13      case !nOKFlag : LoadState(ConfirmDest());
14                  // 承認を得ていなければ問い合わせ処理をプッシュ
15    }                      // SubGoal指定終了
16    Catch : {              // Catch条件
17      case IsPlace(Notified) : // 入力がPlace型ならばCatch
18        Dest = Notified;     // Catchした入力をDestに代入
19    }                      // Catch条件指定終了
20    OnGoal : {             // 全サブゴールを解決した時の処理
21      ExecSetDestination(Dest); // アプリの目的地設定機能を呼び出し
22      ReadText(Dest+"を目的地に設定しました。");  // 読み上げ
23    }                      // OnGoal指定終了
24  }                        // SetDest指定終了
25
26  State SetDest :: ConfirmDest(){       // SetDestの問い合わせ処理
27    Initialize : ReadText(Dest+"を目的地に設定しますか？");
28    Catch : {
29      case Notified == YES : nOKFlag = TRUE;        // 承認を得た
30      case Notified== NO : Dest = φ;                // やりなおし
31    }
32  }
33
34  Scenario Address { // Addressの対話シナリオ
35    SubGoal :  … // Pref,Cityが指定されたら(詳細略)
36    OnGoal : {
37      Place Result = ExecSearchAddress(SetPref,SetCity);
38                         // アプリの住所検索機能呼び出し
39      ReturnDialog Result; // 検索結果の話題出力生成指定
40    }
41  }
```

図 10 対話スクリプトの例

話題 DB にはスクリプト(図 10)で記述したシナリオを格納する．スクリプトは各話題別のシナリオ(6-41 行目)・文法(省略)を定義するように記述することが可能であり，それぞれの話題について **Catch** 条件(16-19,28-31)や話題出力指定(39)を記述する．また，各話題について対話の焦点となっている情報を優先的に **Catch** するために部分的な話題情報(26-32)を作成してプッシュ(13)するように指定できる．対話管理部はこの話題 DB を参照して動作する．

入力解釈部は話題 DB から文法情報を参照してユーザ入力が該当する文法に基づき話題を推定し, ユーザ入力を内部表現に変換する．その過程で履歴情報を利用した代名詞の解決やフレームの統合といった補完処理を行う．解釈候補が複数存在する場合は話題スタックのトップに近い話題情報に作用する入力を優先的に利用するなどして最上位の解釈結果を対話管理部に通知する.

4. 実験

4.1. カーナビ対話システム

実験システムとして上記のスクリプトを利用してカーナビ対話システムを開発した．このシステムは目的地設定タスクの対話を行い，場所検索（住所，中心点周辺のジャンル検索，住所＊ジャンル検索），場所のプロパティ(詳細説明等)問合せ, 目的地設定, 地図操作など全 9 種類の話題を扱う.

4.2. 実験

カーナビ対話システムの使用経験がない 5 名の被験者にシステムと対話をしてもらった．予め話題変更入力を取り扱うことが可能なシステムであることや取り扱う話題の種類と発話様式について被験者に説明した．また，対話中に参照する発話様式のマニュアルを被験者に渡した.

被験者に依頼した対話のタスクは場所の検索条件を自由に指定し，システムが提示した場所から被験者が最も良いと感じた場所に目的地を設定することである．実験終了後に対話ログから話題変更入力の有無を確認すると共に，被験者が誤動作と感じた箇所(システム発話)を指摘してもらった.

今回の実験は対話方式の評価が目的であるので，音声の誤認識の影響を低減する必要がある．そのためにシステムへの入力方法は被験者の代わりに実験者が音声入力を代行する WOZ 方式を採用した．実験者は被験者の入力をモニタし，同等の内容をシステムの入力文法に沿った音声で入力する．タスク外入力は原則リジェクトとした.

4.3. 実験結果

5 名の被験者のうち 4 名が最も良いと感じた場所に目的地を設定することができた．即ちタスク達成率は 80%である．対話例を図 11 に示す.

	Restart (TT)	**Ignore**	**Restart (TT 外)**	**Push**	**Ignore** (+消去)	Total
入力	59 (27%)	36 (17%)	9 (4%)	114 (52%)	0 (0%)	218
話題出力	8 (4%)	153 (73%)	4 (2%)	0 (0%)	45 (21%)	210

表 1 話題選択処理結果の内訳

被験者との対話中に行った話題選択処理結果の内訳を表 1 に示す．表 1 において網掛け部分が話題選択処理で話題スタックが更新される場合である．発話入力の 56%が直接的に話題変更を示唆するものであった．また，間接的な話題変更によ

り話題出力の 23%が不要話題の消去(b)など話題スタックに影響を与えている．従って実際の対話において話題選択処理が必要な状況は多数存在していると言える．

図 11 カーナビ対話システムの対話例

今回の実験結果には入力・話題出力についても **TT** 外の話題情報を指定する **Restart** 判定が少なかった．これは全話題の構造が目的地設定(場所を **Catch**)－場所検索を主とした単純なものであることと，対話の流れが目的地設定を **Push** しない状態で先ず場所検索を行うパタンが多く，**Catch** 条件として場所を待ち受ける状態が少なかったためであろう．話題出力を **Ignore** すると共に不要話題の消去をするパタンが多いことも場所検索の話題が複数 **Push** されていたためと考えれば場所を **Catch** できなかった状態が多いことと符合する．

被験者が誤動作と感じた箇所の誤動作要因を調査した．その結果，誤動作要因の大半(28 回)はタスク外入力等の音声入力に関連した項目にあった(表 2)．話題選択処理は入力時，話題出力時の 428 回実行されたが，誤動作の要因となったものは 4 回(0.9%)であり，その主な理由は **Catch** 条件を広めに設計したためであった．以上より提案方式は柔軟に話題選択処理を実行していると言える．

音声入力 (タスク外等)	入力 解釈	話題内 対話方略	話題選択	Total
28 回	2 回	10 回	4 回	44 回

表 2 誤動作要因の分布

実験システムのスクリプトに記述した **Catch** 条件は 42 種類である．これに対して話題選択処理で **TT** 以外の話題を選択したユーザ入力は 35 種類であった．話題選択処理がない対話システムで，今回の実験で行われた対話を受理するためには予めシナリオに 35 種類の分岐条件を追加しなければならず，検討すべき分岐条件は 1.8 倍になる．従って話題選択処理を行う提案方式は対話シナリオの設計コストの低減に寄与していると言える．

5. まとめ

本稿ではユーザが話題を変更する話題変更入力に対して動的に話題選択処理を行う対話方式を提案した．提案方式は全話題間の遷移手順が不要であり，話題別の対話シナリオを参照して対話を進行する．また，ユーザ入力時の直接的な変更，話題終了時に判明する間接的な変更に対応するために両方のタイミングで話題選択処理を行う．

提案方式をカーナビの目的地設定タスクに適用し，対話実験を行った．実験結果によりユーザ発話の 56%が直接的に話題変更を示唆し，話題終了時の 23%の事例に間接的な話題変更による影響があり，話題選択処理が必要な状況が多数存在することを示した．そのような状況下において被験者が誤動作と感じる要因となった話題選択処理は全体の 0.9%であり，提案方式の動的な話題選択が対話の柔軟性向上に寄与することを示した．

今後は各話題における協調応答生成や対話方略のテンプレート化[4]を検討する予定である．

[参考文献]

[1] D.Goddeau, H.Meng, J.Polifroni, S.Seneff, S.Busayapongchai, “A Form-Based Dialogue Manager for Spoken Language Applications”, ICSLP ‘96(1996)

[2] J.Chu-Carroll, “Form-Based Reasoning For Mixed-Initiative Dialogue Management in Information-Query Systems”, Eurospeech’99 (1999)

[3] S.Bennacef, L.Devillers, S.Rosset, L.Lamel, “Dialog in the RAILTEL Telephone-Based System”, ICSLP’96 (1996)

[4] 鈴木夕紀子, 池ヶ谷有希, 野口靖浩, 伊藤敏彦, 小西達裕, 伊藤幸宏, 高木朗, ”モジュラリティの高い対話制御ルールの設計とその具体的な対話ドメインへの適用方法に関する研究”, 人工知能学会研究会資料, SIG-SLUD-A303-12(2004)

[5] D.Bohus, A.I.Rudnicky, “RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda”, Eurospeech’03 (2003)

[6] B.Pakucs, “Towards Dynamic Multi-Domain Dialogue Processing”, Eurospeech’03 (2003)

[7] 奥智岐, 西本卓也, 荒木雅弘, 新美康永, ”タスクに依存しない音声対話の制御方式”, 信学論, vol.J86-D-II, No.5, pp.608-615 (2003)

[8] 屋野武秀, 笹島宗彦, 西山修, 上原龍也, “混合主導対話方式 AMIDAS とカーナビ対話システムへの応用”, インタラクション 2003(2003)