Voice User Interface Design 概要

1. 会話が成立する前提

皆さんは日々誰かと何らかの会話をしていると思いますが、その難しさもまた日々感じていることかと思います。特に、雑談ではなく、適切に会話を組み立てて話し続けることは、誰であってもかなり難しいことです。

単なる雑談ではなく、適切な会話を組み立てて話し続けるのは、かなり難しいこと

例えば、「天気は？」と誰かに聞いたとします。聞かれた側は、何らかの返事をすると思いますが、単に「天気は？」と聞かれた際に、実は頭の中でいくつかの疑問を誰しもが思い浮かべることになるでしょう。

つまり、「天気は？」というシンプルな問いかけでは、回答するために必要な情報が欠如しています。具体的には、「いつの天気を聞いてるの？」や「どこの天気を聞いているの？」などの疑問を思い浮かべることでしょう。

ただし、「天気は？」と聞いてきた相手に特に何らかの疑問をぶつけなくても、回答をある程度決定することができます。例えば、同じ家に住んでいる夫や妻、あるいは子供に対して、朝起きた後に「天気は？」と聞いたのであれば、「その家の場所の今日の天気を聞いてるんだな」と判断して、「今日は晴れだよ」と返事をする可能性が高いはずです。一方、会社の同僚に職場で「天気は？」と聞いても、「どこの？いつの？」と逆に質問される可能性のほうが高いでしょう。

このように、会話は「状況」が非常に強く影響します。この状況のことを「コンテキスト」と呼びます。このコンテキストは、会話を実際に行っている2者が意識していないかもしれませんが、絶対に存在します。そして、会話の中で明示的に2者間がそのコンテキストを全て伝えることも難しいことです。

コンテキストは、例えば具体的に以下のようなことから成り立ちます。

- 話の流れ
- 話し相手のこと
- 今置かれている状況

- 必要とされている事項

会話には必ずコンテキストが存在し、そして会話を進めていくためにコンテキストは絶対に必要な要素なのです。

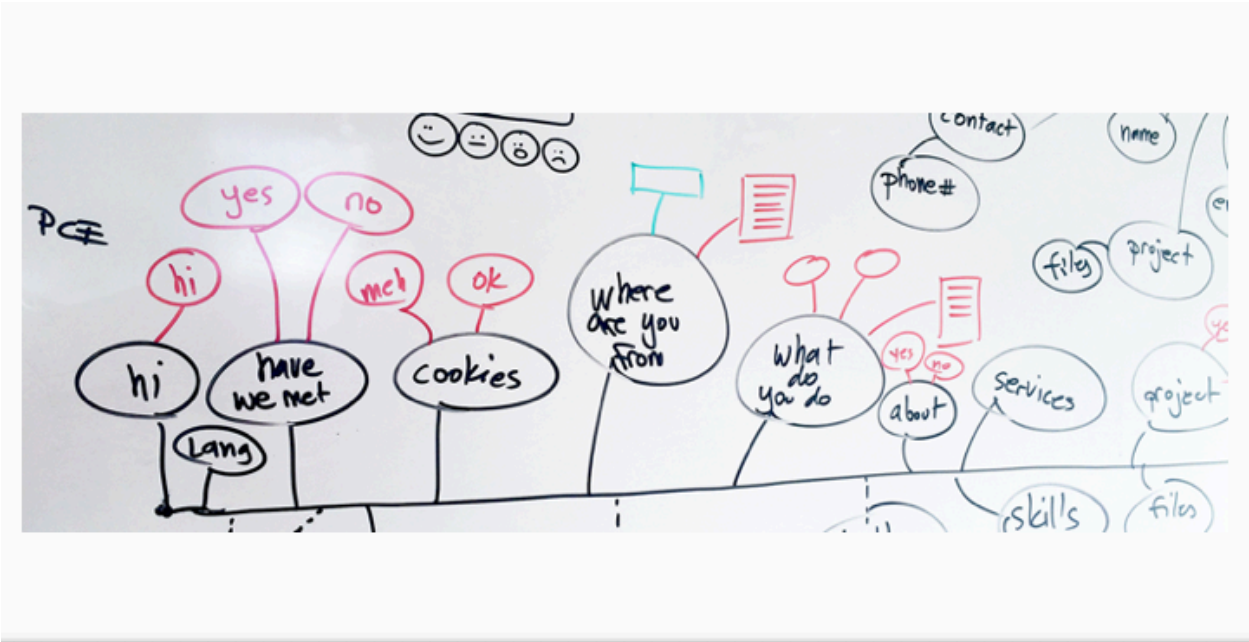
コンテキストは、会話のその場のみで意識されるだけで、会話が終わった後は忘れてしまっても良いことも多くあります。多くの場合は、長時間覚えておくことでもないでしょう。

会話は、コンテキスト次第でうまく行くこともあれば、全くうまく進まないこともあります。会話を成立させるためには、コンテキストを2者間で共有し、その合意形成をしていくことが必要です、そのためには、具体的に以下のような努力を2者で行うことが求められます。

- 役目、目的を会話の冒頭で説明しておく。
- お互いに、役目や目的を欲張らず、シンプルにしておく。
- 会話の理解や言葉の認識がお互い正しくできているだろうと暗黙的に考えることはしない。むしろ、理解や認識がずれている状況、つまり失敗しているだろうということを前提とする。
- 必ずしも相手が期待した回答をしてくるとは限らない、と考える。

2. 会話の流れと組み立て

さて、会話は無秩序なようで、実はそのほとんどが明確な複数の段階に沿って進んでいます。物語に起承転結があるように、会話にも同じような段階が存在しています。



会話は様々な断片に分かれ、基本的にそれらが連続していたりいなかったり。つまり、整理が必要。

言語学者のLeszek Zawadzki氏は、会話にはいくつかの種別があることを提唱しています。具体的には、以下の種別です。

- Opening - 会話のはじめ、きっかけ。
- Extra - 特別な会話、普段ありえない事象。
- Skip - 次の事象、次の会話へ、選択して次へ。
- Core - 核となる大切な会話、主たる内容の会話。
- Chatter - 雑談、会話を楽しむための会話。
- Ending - 会話を終えるための結び、失敗時の終了。
- Help - ヘルプ、手助け、質問。

会話には、始まりがあります。そして、全ての会話には、終わりが来ます。一つの会話の中には、以下の6つのステップがあると考えられています。

1. 共通の土台を据えるためにチャンネルを開く。
2. 約束する。
3. 意味を組み立てる。
4. 発展させる。
5. 同意に集中する。
6. 行動または交流する。

これらをもう少し具体的に言い換えると、以下のようになります。

1. 話し手Aが話し手Bにメッセージを送る。
2. BはAと会話することを約束する。
3. AとBは一連の構造化されたアイディアと文脈を理解し合う。
4. AまたはBが今回のインタラクションから物事を学習し獲得する。
5. 上手くいけばAとBは同意に達する。上手くいかなければ状況の修復を試みる。
6. 会話の結果として実用的なアクションがとられるか、無意識の目標に達する。

誰しもがこのステップに心当たりがあると思います。

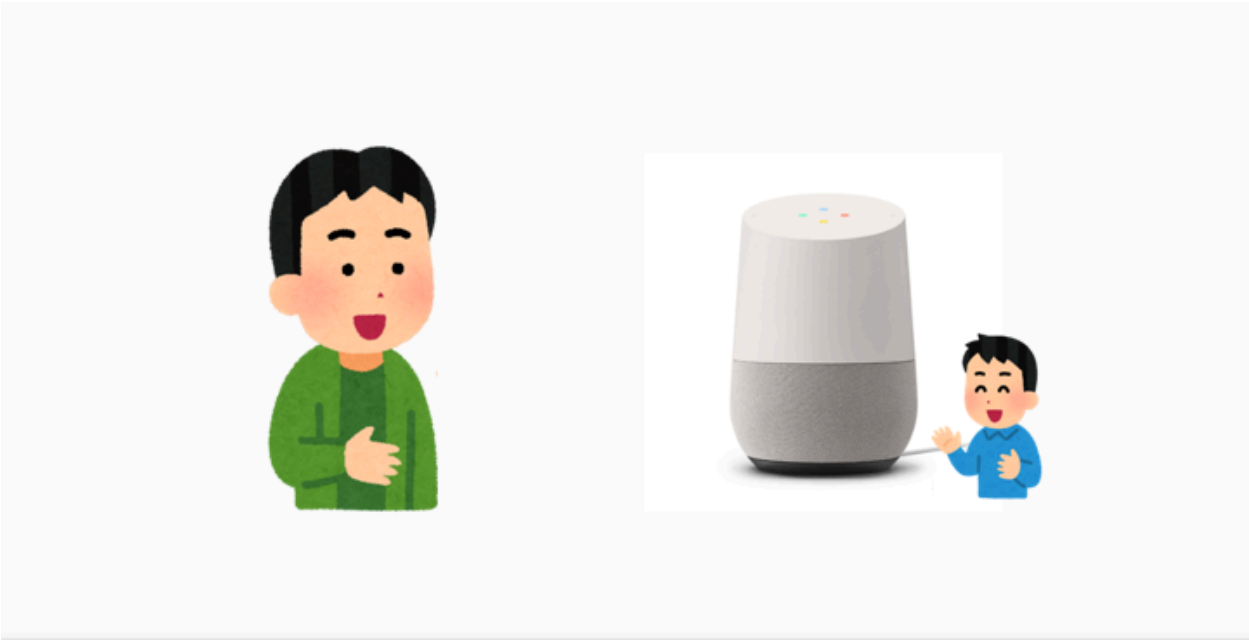
この会話のステップをスムーズに進めるためには、テクニックが必要です。つまり、会話の流れを制御し、そして会話をうまく組み立てるために、以下のポイントが必要です。

- 答えを誘導する - 相手からの返事が「はい」「いいえ、わかりません」のどちらかになるように問いかける。
- 答え方を誘導する - 「何時に起こしましょうか？」と問いかけることで、相手は「7時」といった時間を明確に答える。
- うまく会話を終わらせる - 会話に失敗してうまくいかなかったときも、会話の終了を相手に明確に伝え、納得させる。

3. 会話サービスにおけるキャラクター設定

ここからは、Voice User Interfaceを具体的に考えていきましょう。つまり、会話相手は、人間ではなく、機械である状況です。人は通常、他人に対して何らかの会話をすることで、何かを依頼します。機械が自然言語を理解し、そして自然言語を話せるのであれば、そう、会話相手が機械ではなく、人間であると印象付けられれば、それに越したことはありません。

目標として、会話が人間同士のように自然であることを目指すべきです。



コンピュータを使っている、という感覚ではなく、誰かにお願いしている、という感覚に近い

ここで重要なことは、会話型インタフェースを設計する前に、会話の雰囲気について検討するということです。つまり、それは「ペルソナを定義する」ということです。例えば、以下のようにして雰囲気を検討します。

- 楽しいゲーム - 少し変わった口調を使用する。
- ニュースリーダー - 堅実で改まった口調を使用する。

これら以外にも、名前、性格、特徴、会話のポリシーなどを検討します。

また、もし会話型インタフェースを通じて提供したいものが何らかの既存サービスであり、そのサービスがすでに何らかのブランディングがされているのであれば、その独自のブランドやアイデンティティをペルソナに反映します。これは、ユーザにリピーターになってもらうために、そして会話に一貫性を持たせるために必要なことです。

ペルソナを定義する際に重要な要素として、「距離感」があります。会話相手に対して、「やけに馴れ馴れしいな」や「とても遠慮しているな」といった感想を持ったことがあると思います。ユーザにそのような印象をできるだけ持たせないように工夫すべきです。そのためには、例えばユーザとの距離感を「親しすぎる」と「ぞんざいすぎる」を両端として7段階で定義し、ペルソナの定義の際にどの段階にするかを決めます。

また、設定した距離感が適切かどうかは、実際にユーザからのフィードバックを得てみないと確認できないことです。そのため、日々の調整が、より良い会話への改善に繋がります。

4. おもてなしと信頼の作り方

会話相手の印象は、その相手の身なりや表情、その相手がいる環境などによって大きく左右されます。そして、会話が進んだとしても、その印象が会話に大きく影響します。つまり、ユーザを「おもてなし」するためには、最初の印象が非常に重要です。

実際に人間と話すことや、Graphical User Interfaceと違い、Voice User Interfaceでは視覚情報がありません。そのため、「最初の印象」とは、会話の始まりがどういったものであったかが該当します。

この最初の印象を良くするために、会話の最初として何を言うべきかを検討することが求められます。具体的には、主に2点に気をつけます。

最初のポイントは、「あなたが誰であるかを伝える」ということです。つまり、自己紹介を行います。会話相手がわかることで、ユーザは安心します。

しかし、ユーザに理解させようとするあまり、非常に長い自己紹介は、印象を非常に悪くすることでしょう。次のポイントは、「適量の情報を伝える」ということです。これは、具体的には以下のように言い換えることができます。

- 多くの情報を与えてユーザを圧倒しない。
- 短く、親しみやすい挨拶にする。
- 上級者（すでにこの相手との会話を経験しているユーザ）に対しても同様とする。

会話の始まりだけでなく、会話の終わりもユーザへの印象としては重要です。特に、最後のやり取りがユーザにとって強く印象に残る可能性が高く、適切に会話が終わるかどうかは慎重に検討する必要があります。

具体的には、ユーザがシンプルなフレーズを使って会話を終了できるようにしておきます。しかも、「会話のどの時点でも終了できるようにしておく」ことが大事です。

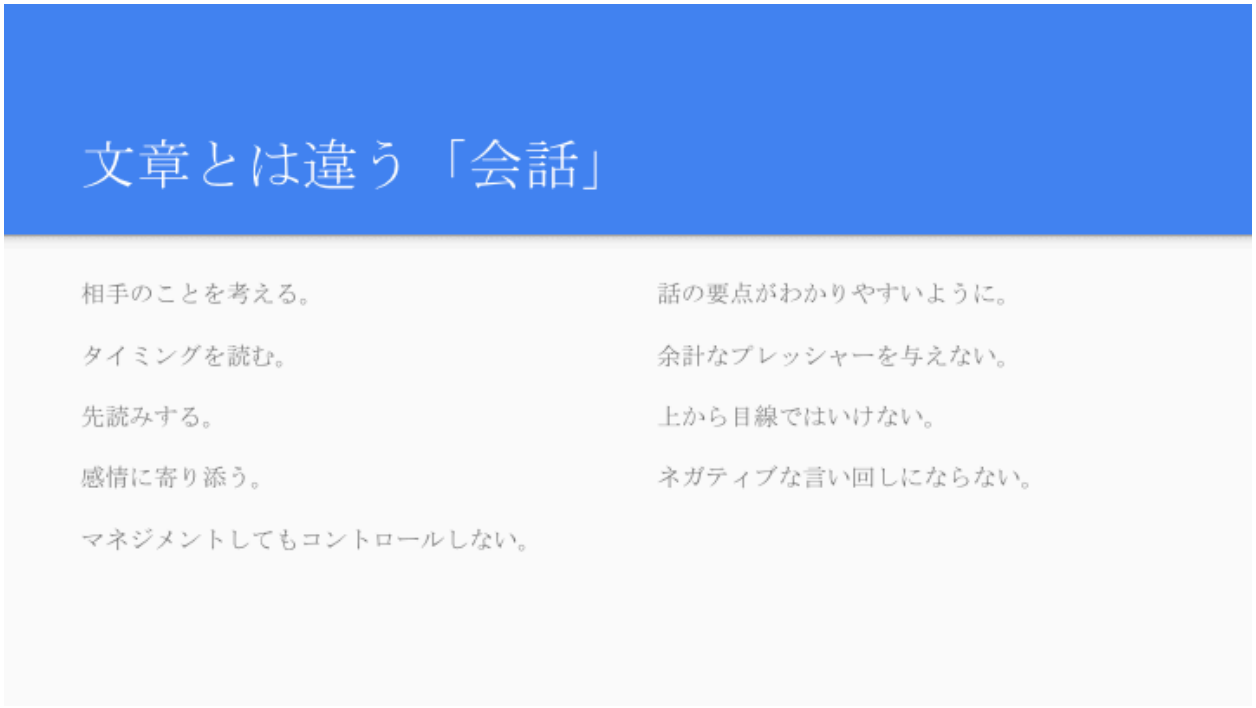
5. 会話としてやり取りする方法、原則

ここではもっと具体的に会話というものを考えていきましょう。ちゃんと成立している会話には、そのための原則があります。

まず、会話は「文章」とは明らかに異なるものです。文章を読み書きする際にはほとんど必要としないことが、会話ではいくつも求められます。ぱっと思いつくだけでも、以下があげられるでしょう。

- 相手のことを考える。
- タイミングを読む。
- 先読みする。
- 感情に寄り添う。
- マネジメントしてもコントロールしない。
- 話の要点がわかりやすいように。
- 余計なプレッシャーを与えない。
- 上から目線ではいけない。
- ネガティブな言い回しにならない。

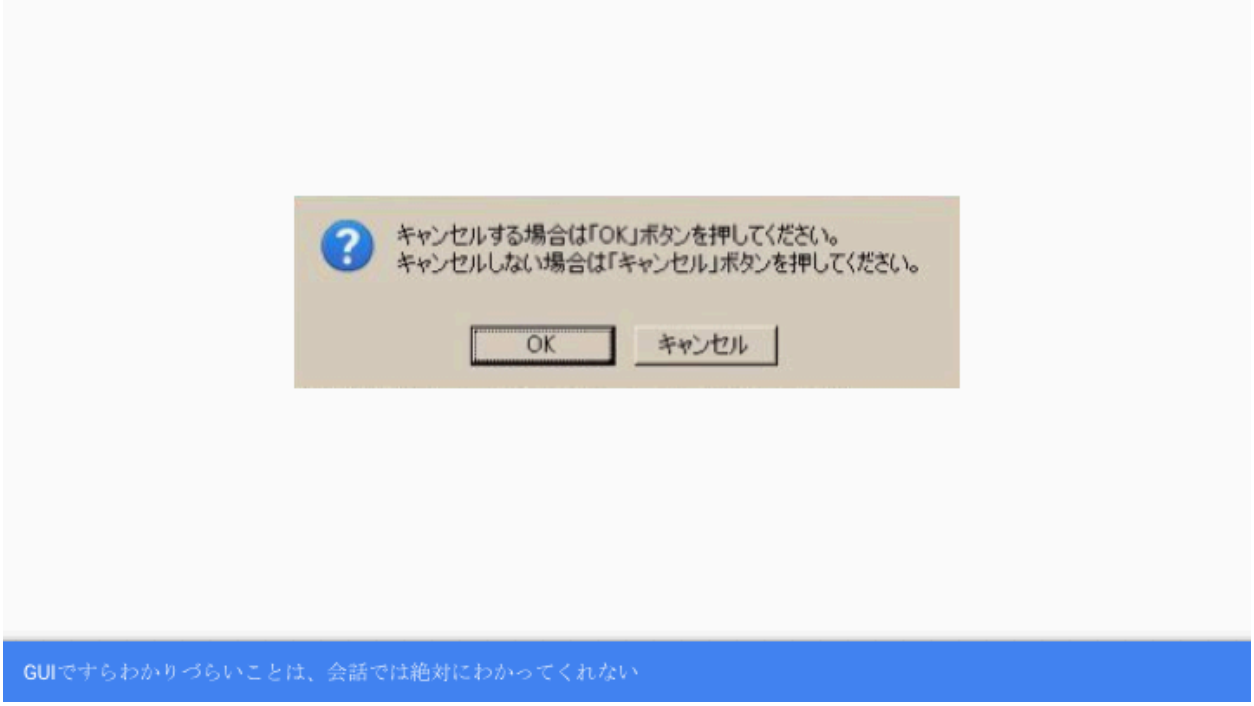
これらのことを会話では「リアルタイムに、そして臨機応変に」行うことになるのです。



一つ単純な例をあげてみましょう。以下のようなダイアログに出くわしたことはありませんか？

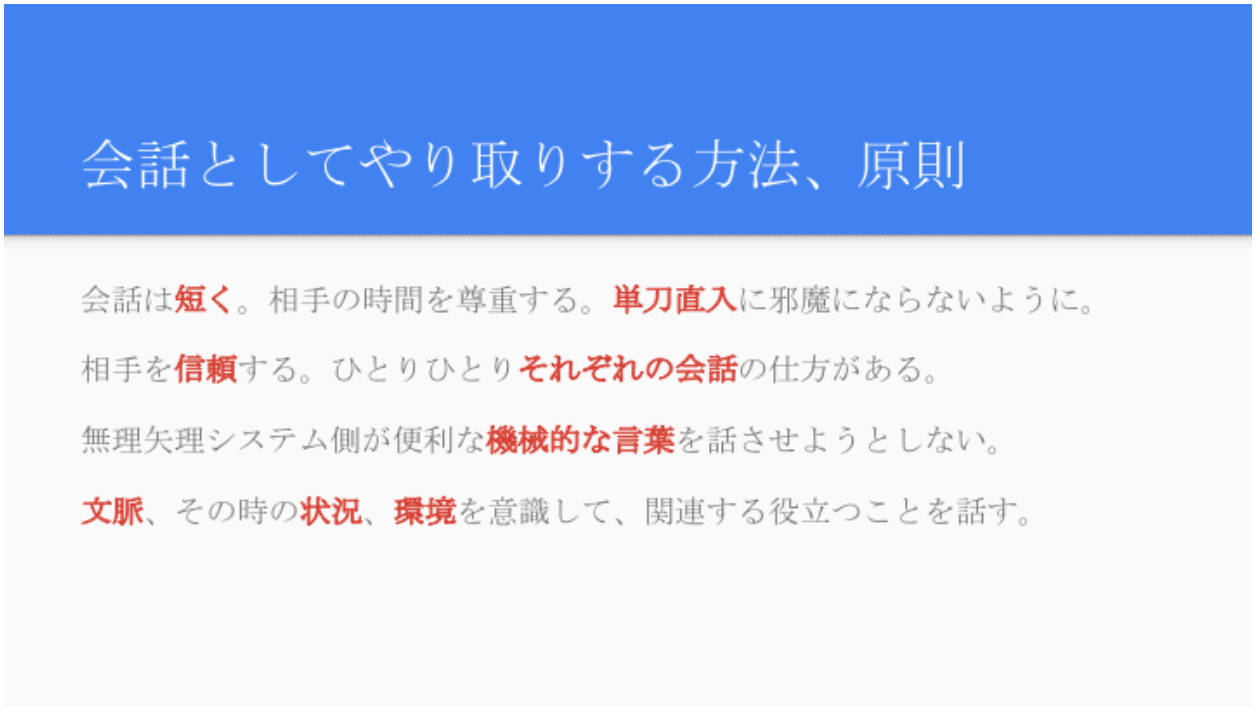
”キャンセルする場合は「OK」ボタンを押してください。キャンセルしない場合は「キャンセル」ボタンを押してください。”

一体どちらのボタンを押すことが正解なのか、すぐにはわからないと思います。Graphical User Interfaceの場合は、文章を何度も読み返すことで、その意図を理解できるかもしれません。しかし、Voice User Interfaceでは、一度しか聞けないことがほとんどです。そのため、この難解な文章を1回聞いただけで理解できた人のみが、正しい選択をすることができます。これは、決して良いUser Interfaceとは言えないことは明らかです。



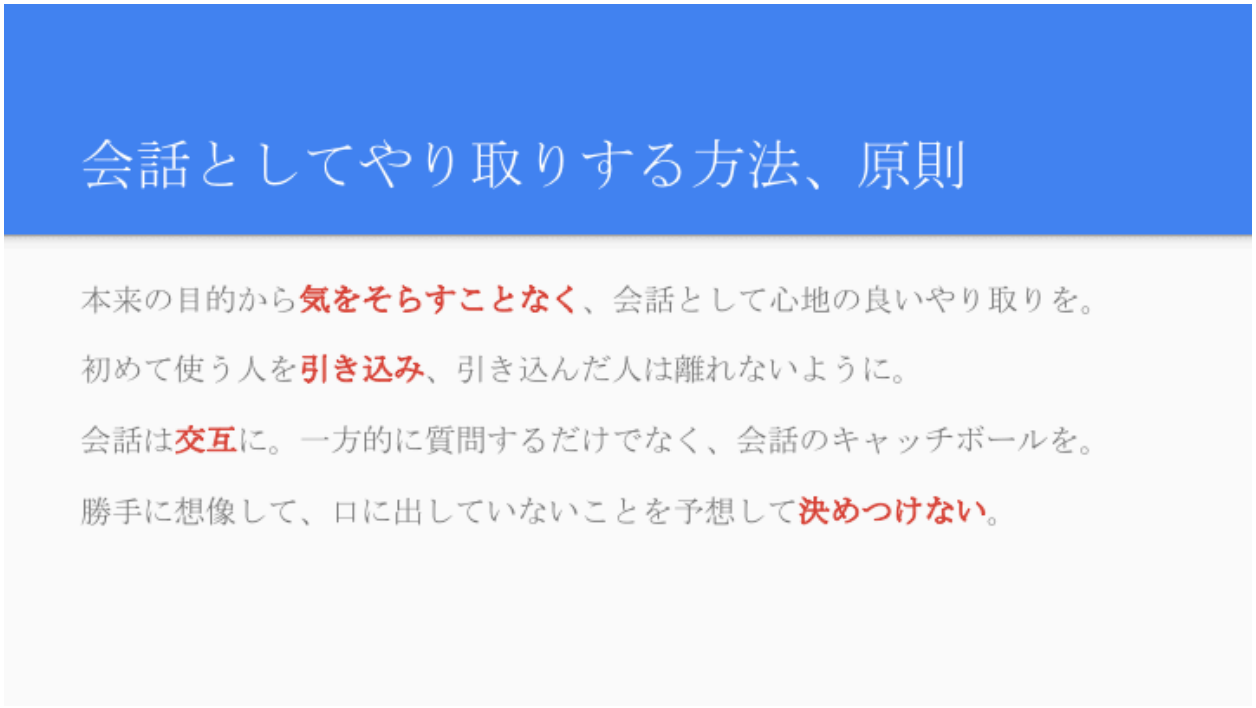
では、より良い会話となるために工夫すべき点について、詳しく見ていきましょう。これらは、会話型インタフェースを設計するための原則と言えるものばかりです。

- 会話は短く。相手の時間を尊重する。単刀直入に邪魔にならないように。
- 相手を信頼する。ひとりひとりそれぞれの会話の仕方がある。
- 無理矢理システム側が便利な機械的な言葉を話させようとしない。
- 文脈、その時の状況、環境を意識して、関連する役立つことを話す。



- 本来の目的から気をそらすことなく、会話として心地の良いやり取りを。
- 初めて使う人を引き込み、引き込んだ人は離れないように。
- 会話は交互に。一方的に質問するだけでなく、会話のキャッチボールを。
- 勝手に想像して、口に出していないことを予想して決めつけない。

これらは、どんな会話においても、適用できる原則となります。



ペルソナの定義の話の中で、如何にユーザに自然な会話と印象づけるか、つまり会話相手として、できるだけ機械ではなく人間として認識してもらうための工夫をすべきだという話をしました。これを言い換えると、自然な言い回し、という表現となるでしょう。

日頃皆さんが会話型インタフェースとして非常に多く体験していることとして、「コールセンターでの自動応答」があると思います。この自動応答に対して、自然な会話だと印象を持つことは全くと言って良いほどないはずです。コールセンターの自動応答においては、できるだけ簡単に、そして間違うことなくユーザに操作を行ってもらうために、「○○したい場合は1を、それ以外の場合は7を押してください」といった操作を自動応答側から指定されます。こんなことは、人間同士の会話ではあり得ないことです。

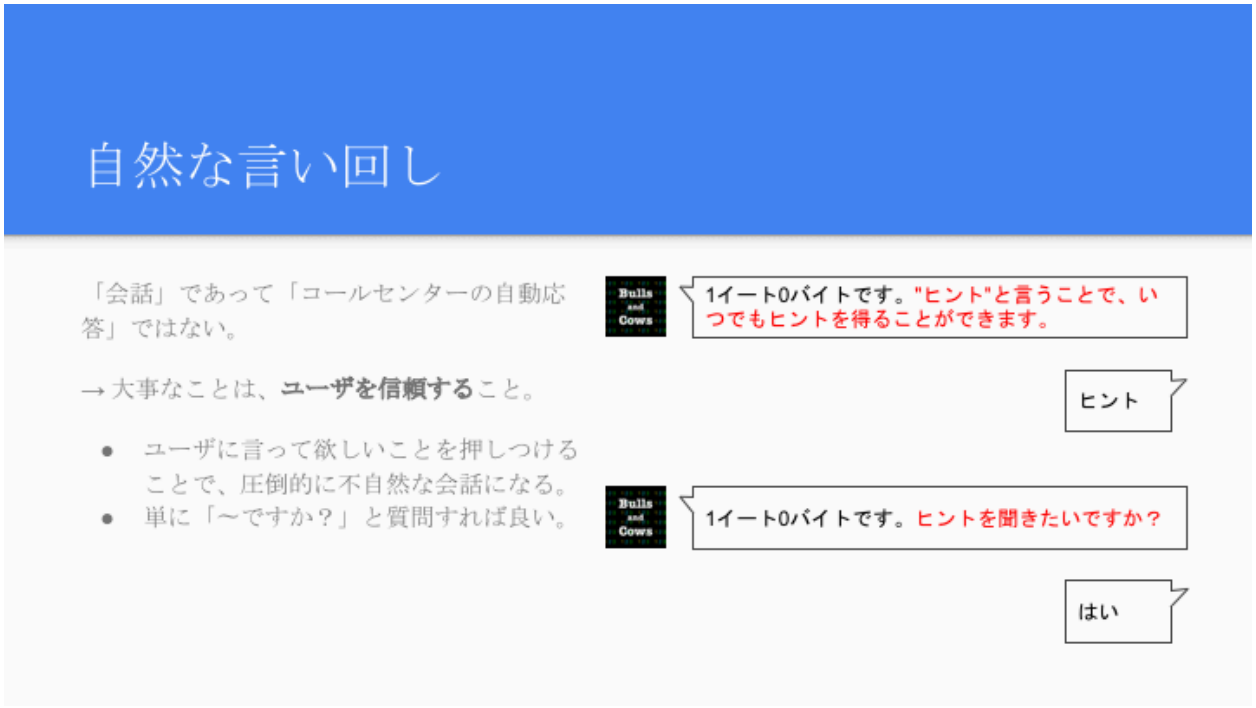
会話型インタフェースを設計する際に、期待する発言をユーザにして欲しいがために、例えば以下のようなことを指定したくなります。

「”ヒント”と言うことで、いつでもヒントを得ることができます。」

この発言は、ユーザを信頼していません。”ヒント”という言葉を押し付けることで、圧倒的に不自然な会話となってしまっています。ユーザを信頼していれば、単に、

「ヒントを聞きたいですか？」

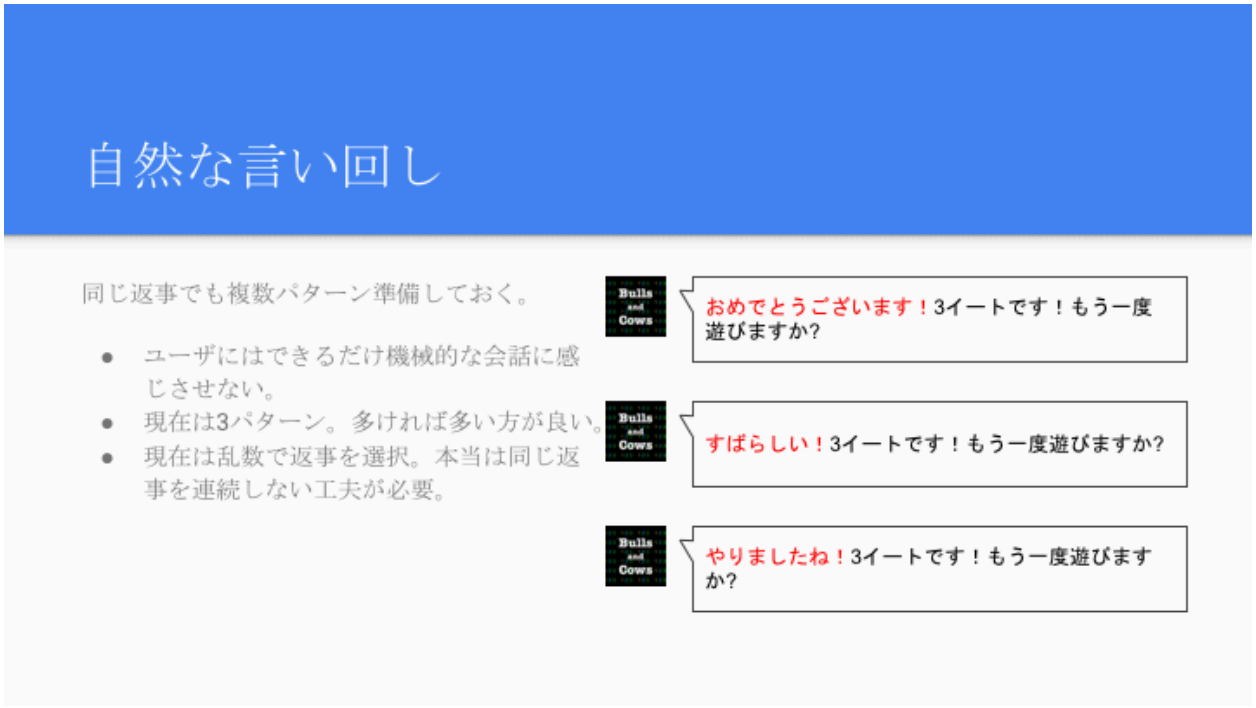
と聞けば良いはずです。ユーザは「はい」もしくは「聞きたくないです」といった返答をするはずであり、このやり取りは自然です。



もう一つ自然な言い回しを実現するための工夫があります。ここで、如何にも機械的だな、と感じてしまうシチュエーションを想像してみてください。その代表例が、「同じ表現を繰り返す」です。

例えば、数当てゲームにおいて、ユーザが数を当てた際に、「おめでとうございます！もう一度遊びますか？」と毎回同じフレーズを言ってしまうと、これは如何にも機械的です。そうではなく、2度目は「すばらしい！良く当てましたね！また遊びますか？」というように、同じ意味でもその表現を変えることで、機械的な印象は大幅に薄まります。表現のパターンをもっと増やすことで、その自然さは増します。

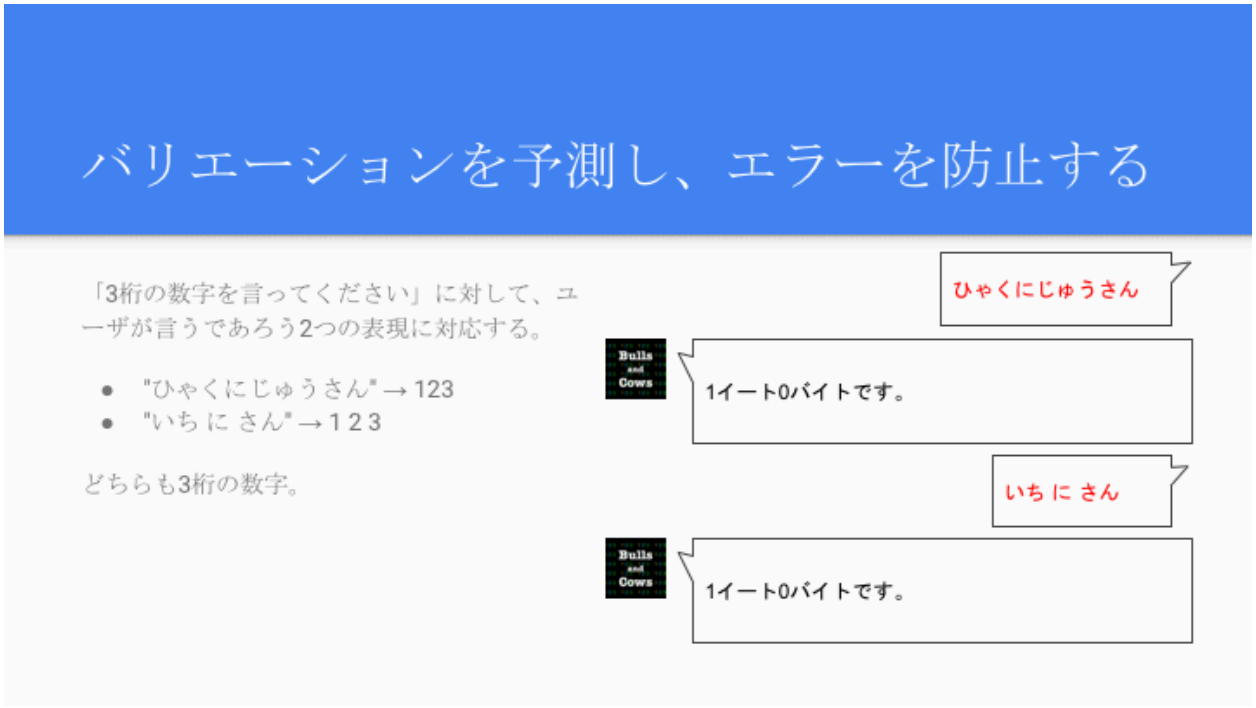
ポイントは、繰り返さないことと、ユーザに規則性を見破られないこと、の2点です。基本的に表現をランダムに選択して、前回と同じ表現が選択されてしまった場合は異なる表現を選択し直す、といった工夫を実装すると良いでしょう。



ユーザは、必ずしも期待する表現をしてくれるとは限りません。例えば、数当てゲームにて「3桁の数字を言ってください」とユーザに要求したとします。123という3桁の数字に対して、ユーザは少なくとも以下の2つのパターンを言ってくると想像できます。

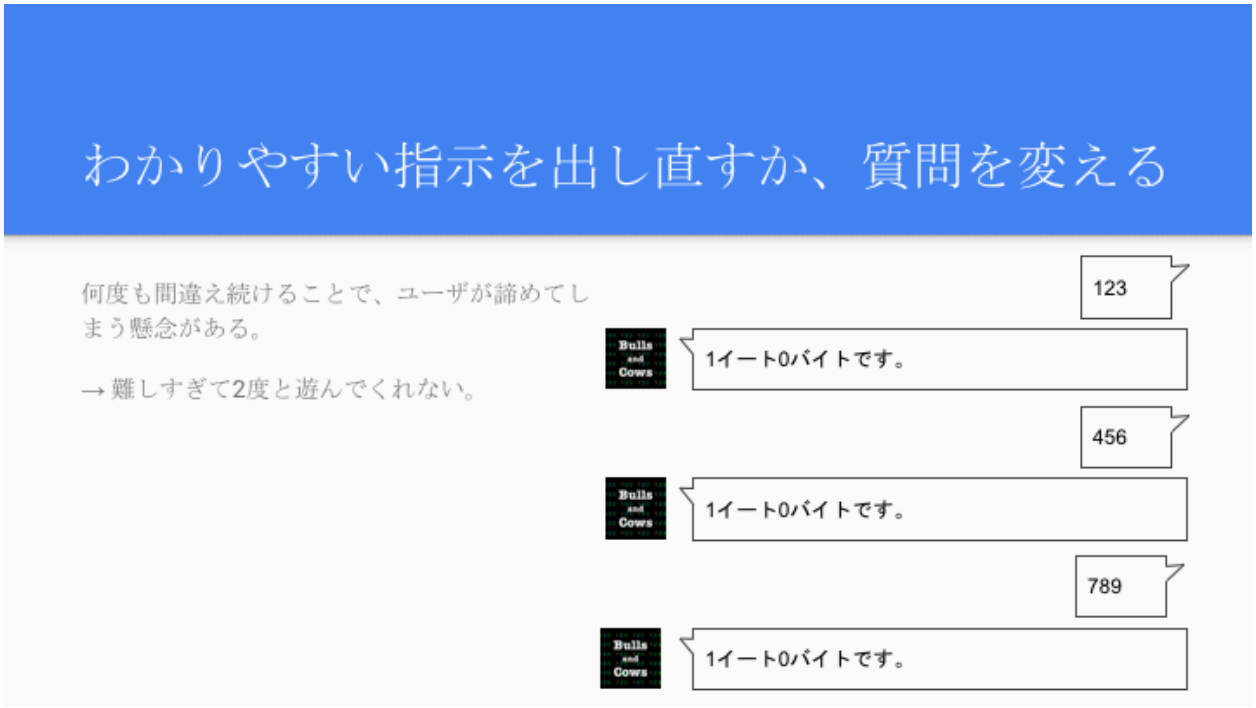
- ひゃくにじゅうさん
- いちにさん

どちらも3桁の数字を読み上げていることに変わりはありません。これは、両方とも許容されるべきであり、片方がエラーとなってしまうと、ユーザに大きなストレスを与える結果となってしまいます。事前に表現のバリエーションを予測して、それらを処理可能にしておく必要があります。これは、会話を継続させる上で、およびユーザの利便性を高める上で、とても重要です。



自然な言い回しを実現する最後のポイントとして、会話を前に進ませるための指示を出す、あるいは質問を途中で変更する、というテクニックがあります。

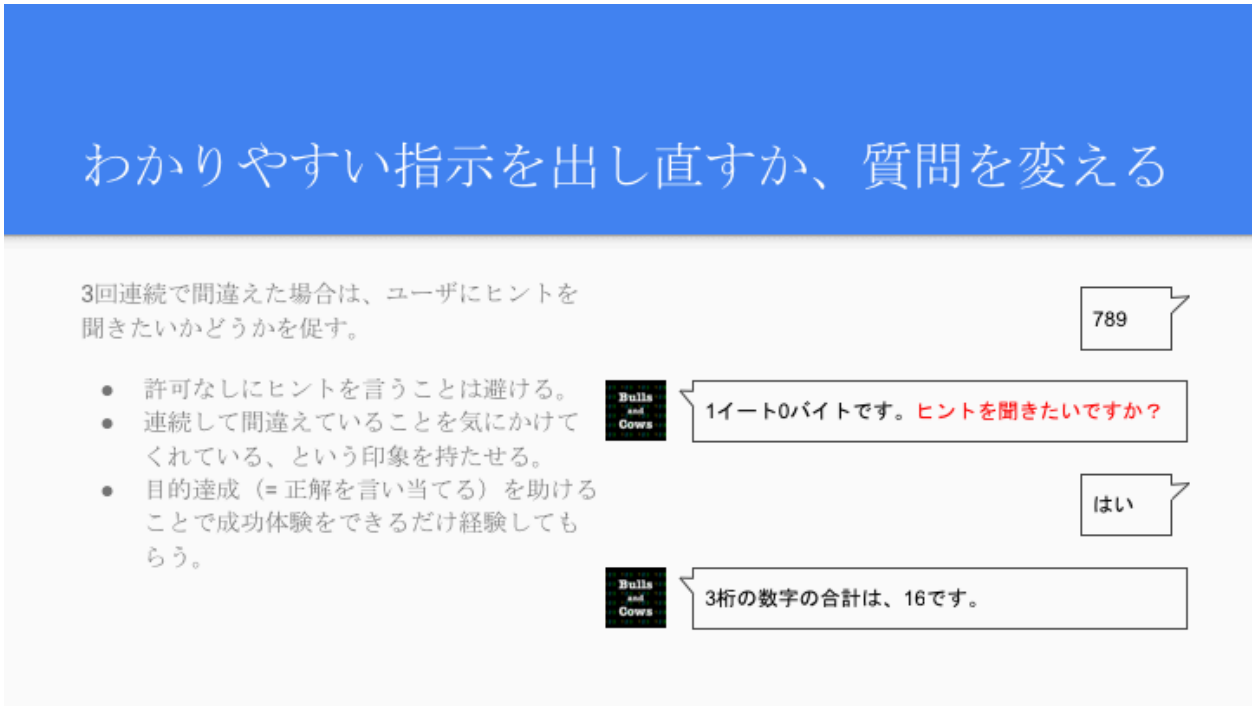
例えば、数当てゲームの場合、ユーザは何とか数を当てようとして、数を言い続けます。それらに対して、「ハズレです」と繰り返し言い続けてしまうと、ユーザは早々に諦めてしまい、難しいと感じて二度と遊んでくれない可能性が大きくなってしまいます。



そこで、例えば「3回間違えたら、ヒントを聞きたいかどうか聞いてみる」という工夫を入れます。このポイントは、以下の3点です。

- ユーザの許可なしにヒントを言うことは避ける。
- 連続して間違えていることを気にかけている、という印象をユーザに持たせる。
- 目的の達成を補助することで、ユーザにできるだけ成功体験を経験してもらう。

こういった工夫が、自然な会話の要素となり、結果として継続的に使ってもらえる会話型インタフェースが実現されます。

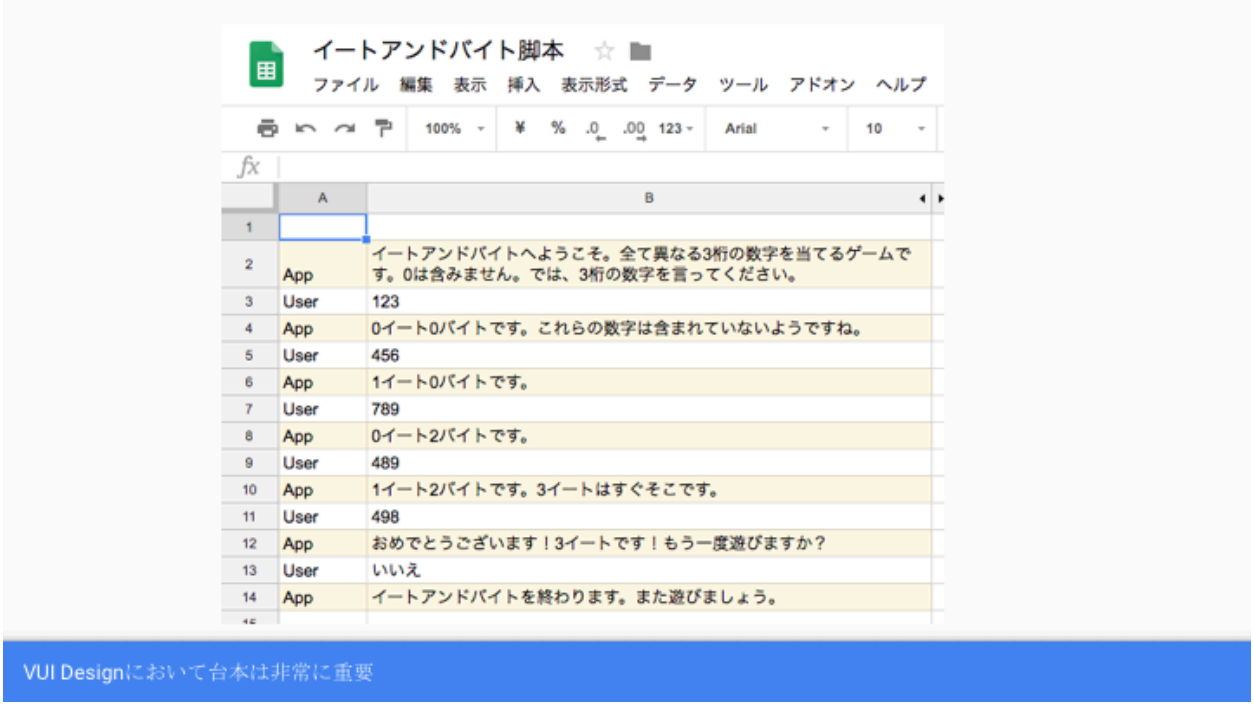


6. 台本の書き方

Voice User Interfaceの設計において、台本は非常に重要です。

ここでの台本とは、ユーザとVUIアクションとのインタラクション、つまり会話のやり取りの具体的な例です。VUIアクションからの問いかけに対して、想定されるユーザの返答を書き、さらにその返答への反応としてVUIアクションが更にどう問いかけるか、その連続が台本となります。

ドラマの脚本、と言い換えても良いかもしれません。



この台本の書き方にも工夫すべき点がいくつかあります。

- 一息で話せる長さ。（声に出して読み上げる）
- 自然なセリフ。（話しやすい雰囲気作り）
- ユーザ発話の方向付け。（質問を投げる）
- 会話の目印を利用する。（「まず」、「それから」）
- 変化をつける。（特にあいさつ系の言葉）

- 言葉の統一。(動詞と名詞はいつも同じ言葉遣いを)
- 過去の発話を覚えておく。(中断と再開)
- 問題が発生した場合の対処。(特に丁寧に)
- 文脈に沿ったヘルプの提供。(なるべく少なく)
- 録音されたオーディオの利用。(時には音源やラジオ)

台本を書く際には、上記の一覧を手元に常に置いておくと、会話型インターフェスにおける良い台本が書けるはずです。

また、音声サービスであることをうまく活かすことも重要なことです。つまり、会話相手が人間ではなく機械であることが、いくつかの利点を生み出してくれます。具体的には、以下のようなことがあげられるでしょう。

- 音声サービスならではの価値を入れ込む。
- 音声サービスには心理的障壁をさげる意味もある。
- 人が応答できない時間帯でも、いつでも対応できる。
- 気軽に聞いて、簡単に答えてくれる。
- 文章ではなく、会話。
- 人間は面倒すぎてやらないことを任せる。

これらを活用することができればできるほど、繰り返しユーザに使ってもらえる機能となるはずです。

7. 試作、テスト

台本が書き上がったところで、本当にうまく機能する会話になっているかどうかを確認したくなることでしょう。ここで、実際に試作し、テストをすることになります。Graphical User Interfaceを伴うアプリであれば、プロトタイプを開発して実際に触って検証する、ということが必要になります。

しかし、会話型インタフェースでは、そういった開発は一切必要ありません。何も作らなくても、会話そのものはテスト可能なのです。その方法は、とても簡単です。台本を使って、二人の人間のうち、一人をユーザ、もう一人をVUIアクションに見立てて、実際に会話をしてみれば良いのです。

これにより、以下の発見があるはずです。

- 足りなかった言葉の発見
- 言い回しの変更
- 欠けている情報の発見
- より適切な用語
- 言葉遣い

人間同士の会話で不自然と感じてしまうのであれば、その台本の内容はもっと改善できるはずです。

ただし、残念ながら、どんなに台本を使って会話を検証しても、所詮は想定した範囲内でのテストしかできません。会話は非常に多様であり、ユーザが何を話してくるか、その全てを予測することは不可能です。

最初から完璧を求めるのではなく、会話型インタフェースにおいては、ユーザがどんなことを言ったのかを毎日把握して、早いタイミングで会話の改善を行っていくことが重要になります。そして、ユーザからの発言がエラーになってしまうのではなく、それを許容することでエラーを回避し期待通りの会話にしていくことを、Graphical User Interface以上に考慮します。そして、ユーザがエラーから抜け出せるようにしていくのです。

この改善を行っていくためには、ユーザが会話のどの時点で離脱してしまったのか、その会話がどこで終わってしまったのか、それを知ることが必要となります。

8. まとめ

ここでは、会話型インタフェース、つまりVoice User Interfaceを設計していく上で最低限知っておくべき事項について紹介しました。より詳細な設計での工夫は、以下の資料が参考になるでしょう。

音声／会話インターフェースのUX（ユーザー体験）の勘所 - 株式会社エクサ 安藤幸央

また、以下のウェブサイトには、Voice User Interfaceを設計するために必要となる前提知識や注意点などが解説されています。

Conversation design - Actions on Google

Graphical User Interfaceに比べて、Voice User Interfaceは、開発自体は簡単です。その代わり、設計の難易度はかなり高く、また現在でもまだ確立された方法論はありません。ユーザもまだ慣れていないこともあり、しばらくは試行錯誤を繰り返していくことになります。それだけこの分野には面白さがあります。

この資料を通じてVoice User Interfaceに興味を持っていただけることを期待しています。

2019/03/21 Google Developers Expert (Assistant, Web Technology) Yoichiro Tanaka