1O2-OS-18-6

データモデル定義に基づくマルチモーダル対話記述言語の設計と実装

Design and Implementation of multimodal interaction description language based on the data modeling.

竹腰大輔[*1] 荒木雅弘[*1]
Daisuke TAKEGOSHI Masahiro ARAKI

[*1]京都工芸繊維大学
Kyoto Institute of Technology

As the spread of various input and output devices, it is necessary to establish a methodology of developing multimodal interactive system. Previous MMI(MultiModal Interaction) description language used a state transition model for writting interaction and that has low maintenancebility and low expansion capability. In this paper, we design and implement a new MMI description language named MrailsScript from Software Engineering point of view. MrailsScript consists of datamodel definition that can be inherited from semantic web class definition, and annotations of interaction task and dialogue initiative. It can generate a prototype of MMI application automatically.

1. はじめに

近年のコンピュータ技術の発展による音声入出力機器やタッチパネル，モーションセンサなど多様な入出力デバイスの一般的な普及により，様々な入出力を組み合わせたマルチモーダルアプリケーションが開発されている．この中で通常の Web アプリケーションにも，マルチモーダルインタラクション (MMI) が可能なシステムが求められると考えられる．

またセマンティック Web 関連技術の成熟が進み，Schema.org[*1]という共通のオントロジーや FreeBase[*2]のような大きな Web データベースが制定された．システムのバックエンドに質の高い情報を利用可能になったことから Web アプリケーションでの MMI が実現しやすくなり，MMI 開発方法論の確立が必要になっていると考えられる．

一方で，これまで提案されてきた MMI 記述言語では状態遷移モデルに基づく対話制御が多く，Web アプリケーションに必要な機能拡張性や保守性に問題がある．またオブジェクト指向的である OwlSpeak[Heinroth 10] も GUI フレームワークとの開発方法論の違いにより，開発者が移行するコストとリスクが大きい．

この問題を解決するため，ソフトウェア工学の観点から Web インタラクション記述言語を見直し，対話記述言語を中心とした MMI システム全体の設計を行う．MMI システムのためのフレームワークとして，ソフトウェア工学において保守性と機能拡張性の高さが確認されているデータ駆動設計や MVC アーキテクチャを取り入れた，Grails[*3]という GUI 用フレームワークを用いる．

本研究では，MMI に対応した対話タスクの分類と主導権，およびセマンティクス記述とデータモデルを対応させ，マルチモーダル対話アプリケーションのプロトタイプを作成可能な対話記述言語，MrailsScript の設計と実装を行った．

連絡先: 京都工芸繊維大学情報工学部門
〒 606-8585 京都市左京区松ヶ崎橋上町
E-mail: takegoshi@ii.is.kit.ac.jp

∗1 http://schema.org/
∗2 http://www.freebase.com/
∗3 http://grails.org/

2. 設計

2.1 MMI アーキテクチャ

MMI システムの設計を行うにあたり，マルチモーダル対話を行うためのシステムと，多様な入出力機器の情報を統合・分化させる環境を構築するためのアーキテクチャがある．前項での既存の MMI システムの問題点から，MMI システム開発方法論の用件として以下の 3 件を重視する．

- MVC の明確な分離
- GUI ベースの Web アプリケーション開発方法論との共通性の確保
- モダリティの変更に対する保守性・機能拡張性の維持

W3C の提案する W3C-MMI-WG アーキテクチャ[W3C 11] では対話管理部とデータモデルの結合が未定義であり，Gralaxy-II アーキテクチャ[Seneff 98] では MVC 記述がハブスクリプトに集中する．これらと比較して，ITSCJ 音声インターフェース委員会で提案されている ITSCJ MMI システムアーキテクチャ[新田 07][ITSCJ 10] は，上記 3 点を満たしよりオブジェクト指向的であるといえ，本研究ではこのアーキテクチャに沿った開発を行った．図 1 に ITSCJ アーキテクチャの構成を示す．ITSCJ アーキテクチャの第 6 層であるアプリケーションロジックとデータモデルには Grails のモデル定義を用いる．第 5 層はタスクと主導権の違いに合わせた MMI 用の Grails コントローラとし，第 4 層は HTML に MMI 用の情報を付加した拡張 HTML の解釈系とした．

2.2 対話記述言語 MrailsScript

データモデル定義を基礎として，インタラクションタスクの設定と主導権の設定をアノテーションで付加し，オントロジーとして採用した Schema.org クラスの継承を行うことでセマンティクスに対応した，対話記述言語 MrailsScript を設計した．

図 2 は設計した MrailsScript によって記述した，本の検索システムの例である．タイトルあるいは著者名による検索を想定している．データモデルで定義するプロパティの意味が Schema.org のプロパティと同じ場合，継承した Schema.org クラスのプロパティ名を使用することでオントロジーに対応す

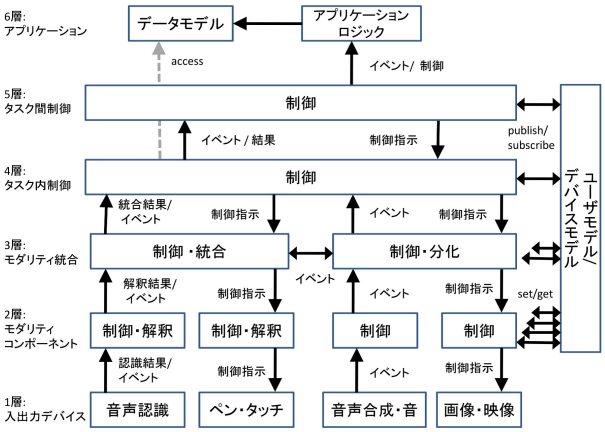

図 1: ITSCJ MMI システムアーキテクチャ

```
1  @SystemInitiative
2  @DBSearch
3  class MyBook extends Book {
4    int price
5    static constraints = {
6      name( onsearch:"ilike" )
7      author.name( onsearch:"ilike" )
8      thumbnailUrl( )
9      price( )
10     isbn( )
11   }
12 }
```

図 2: MrailsScript の例

る．図 2 の name プロパティから isbn プロパティまでは Book クラスのプロパティであり，制約規則内でのプロパティ名の記述がプロパティ宣言を兼ねるものとした．Book クラスの url プロパティであれば，型が String，制約が url:true のように，Schema.org クラスのプロパティ定義と Grails の型・制約を対応させた．プロパティ型の制約は，一般的なデータ型を網羅しており汎用性もある，HTML5 の input タグの type 属性と同様の分類によって指定する．

図 2 の 7 行目のように author プロパティが別 Schema.org クラスのインスタンスである場合，インスタンスのプロパティを”.”でつないで指定する．

2.3 対話タスクの分類

Web 上でのインタラクションではシステムとユーザ間の情報のやりとりによってタスクが遂行される場合が多いため，情報のやりとりの仕方によってタスクを 3 つに分類し，データモデルとタスクを対応させた．

- @Explanation 説明タスク

  システムからユーザへ情報を提供するタスク．通常の Web ページの閲覧や，ユーザにチュートリアルを教示する場合などが当たる．

- @Slotfilling スロットフィリングタスク

  ユーザがシステムへ情報を入力，提供するタスク．アカウント取得のための個人情報の入力など，入力項目を埋めていくようなタスクが当たる．

- @DBSearch 検索タスク

  ユーザがシステムへ情報を入力し，システムから何らかの結果を返してもらうタスク．Web ページの検索やトラブルシューティングなどが当たる．

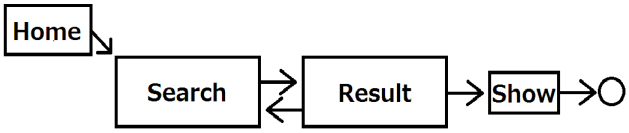

図 3: データベース検索タスクの遷移図

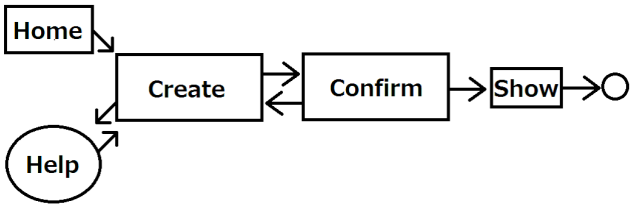

図 4: スロットフィリングタスク (新規作成) の遷移図

以上のタスクをアノテーションで指定し，それぞれのタスクで使用する条件をデータモデル定義中に記述する．検索タスクでは図 2 の 6 行目と 7 行目のように，検索に使用する要素と制約条件を onsearch 属性として追加し ”ilike”(部分一致), ”eq”(全一致) など検索基準を指定する．スロットフィリングタスクでは入力が空であることを許す nullable 属性に真偽値を指定する．今回は典型的な 3 つのタスクを実装するため，タスクのフロー遷移をそれぞれ図 3 と図 4，説明タスクは単純な要素の表示のみであると定義した．

2.4 主導権の分類

対話システムでは，達成するべき目標や想定されるユーザによって対話の流れを制御する主体，つまり対話の主導権を持つものが変化する．

- @SystemInitiative システム主導

  システムが決められた手順に沿って，ユーザに順次情報の入力を求めることでタスクを達成する形式．

- @UserInitiative ユーザ主導

  ユーザがシステムに自由に情報を入力し，システムがその入力に対して情報を返すことで，タスクを達成する形式．

- @MixedInitiative 混合主導

  ユーザとシステムの間で主導権が入れ替わりつつタスクを達成する形式．

タスクの場合と同様に，データモデル中にそれぞれの主導権で使用する条件を追加する．混合主導では，入力要素の一部が空であった場合の動作などをデータモデル定義に記述する．

2.5 オントロジー

データモデルのオントロジーとして，Schema.org を採用し，Schema.org のクラスを継承してその語彙を自由に使用できるようにした．

Schema.org は検索大手 Google，Yahoo!，Microsoft の 3 社が共同で制定したオントロジーであり，HTML5 で策定された microdata によるマークアップ方式を採用し，広範な語彙を持ち Web 上の多くのデータをマークアップできる．また簡便な拡張メカニズムを持ち，仕様に従った拡張を行えば解釈系が適切にデータを理解して利用できる．検索大手 3 社が提供し今後検索に利用すると表明したことで開発者が Schema.org を選択する大きな理由ができ，この 1 年で Web 上に Schema.org でマークアップされたデータは増大した．言語レベルでオントロジーに対応し，また生成されるビューコードのマークアップを自動生成を行うことでセマンティック Web との親和性を高めた．

3. 実装

Grails フレームワークを使用し，MrailsScript で指定された要素を反映した，Grails のモデル・ビュー・コントローラを生成する処理系を実装した．GUI 記述へ音声モダリティでの入出力機能を付加したものを MMI 記述と位置づけ，音声入出力を実装した．

3.1 データモデル定義の生成

データモデルは，MrailsScript から Grails 標準のデータモデルを生成した．対話記述であるタスクと主導権のアノテーションを取り除き，継承した Schema.org クラスのオントロジーに基づきフィールドと制約を自動生成する．

3.2 コントローラコードの生成

コントローラコードでは，ビューを正しく制御するために必要なアクションを生成する．タスクのフロー遷移図，図 3，図 4 に従って，MrailsScript のタスク・アノテーション指定からコントローラコードとビューコードを実装した．データベース検索タスクでは，検索，検索結果の一覧，詳細の順に画面が遷移する．検索画面では，対話記述の制約条件中の onsearch 属性に基づいて，ビューから送られたリクエストをデータベースへの検索クエリに使用する．

3.3 ビューコードの生成

MMI を行うための要件として，ビューコード内でタスクの流れとは別に主導権を設定できる必要がある．また入出力モダリティ毎にビューコードを用意するのではなく，モダリティ独立なビューコードを元に，その構造を変更することなくモダリティ依存な情報を付加することで，モダリティの拡張が可能であることが望ましい．本実装では，HTML5 の要素を拡張し音声インタラクション可能にしたものを MMI に対応したビューコードと位置づけて，主導権設定と音声モダリティ依存の情報を付加したビューコードを生成した．

図 5 は図 2 のデータモデルより生成された，Grails が解釈可能な GSP コードの一部である．

通常の HTML タグに加えて図 5 の 4 行目のような if 分岐，5 行目の${flash.message} のような文字中式の記述，17-18 行目の g:textfield タグのようなテキスト入力要素を生成する GSP タグの記述が可能で，実行時に解釈され図 5 から動的に図 6 の HTML コードが生成される．

音声対話への対応として，HTML5 で新たに実装された input 要素の x-webkit-speech 音声入力属性を使用した．また label 要素に追加した tts 属性を，入力情報の説明を記述している音声出力要素とした．tts 属性は平文で記述され，各入力要素にフォーカスが移ったとき図 5 の 7 行目の PlayTTS 関数によって，GoogleAPI と JavaScript で実装された音声合成器に送られる．また 3 行目はシステム主導対話での冒頭の対話であり，システムが対話の主導権をとるために，各画面に遷移した際に自動で発話される．

ユーザ主導対話は Web の GUI 操作と同様の手段で入力するスロットを選択するような実装を行った．

```
1  <h1>
2  <g:message code="default.search.label" default="Search" />
3  </h1>
4  <g:if test="${flash.message}">
5  <div id="opmessage" role="status">${flash.message}</div>
6  </g:if>
7  ...
8  <form method="post" action="/sample/myBook/search">
9   <fieldset class="form">
10   <div class="fieldcontain " ondblclick="PlayTTS('nametts')">
11    <label id="nametts"
12       tts="${message(code:'default.tts.message',
13       args:[message(code:'myBook.name.label',
14       default: "Name")] )}" >
15     <g:message code="myBook.name.label" default="Name" />
16    </label>
17   <g:textField name="name" value="${myBookInstance?.name}"
18       x-webkit-speech="true" lang="${local}" />
19   </div>
20   ...
21  </fieldset>
22  </form>
```

図 5: 生成される GSP コード

```
1  <h1>検索</h1>
2  <div role="status" id="opmessage">
3    検索を開始します
4  </div>
5  ...
6  <form method="post" action="/sample/myBook/search">
7   <fieldset class="form">
8    <div class="fieldcontain" ondblclick="PlayTTS('nametts')">
9     <label tts="タイトルを入力してください" id="nametts">
10     タイトル
11    </label>
12    <input type="text" id="name"
13       x-webkit-speech="true" lang="ja"/>
14   </div>
15   ...
16  </fieldset>
17  </form>
```

図 6: 実行時に動的に生成される HTML コード

```
default.search.label=検索
default.tts.message={0}を入力してください
myBook.label=本
myBook.name.label=タイトル
myBook.authorName.label=タイトル
myBook.price.label=価格
myBook.isbn.label=ISBN コード
myBook.search.opening.message=検索を開始します
...
```

図 7: 国際化コード

3.3.1 国際化

MrailsScript と生成される GSP コードからは，対話記述部分を含めて特定言語に依存した情報が除かれている．コンテンツ，本の検索システムであれば本の DB が多言語化されていれば，国際化コードを機械翻訳 API によって他の言語に翻訳して設定することで，簡単に多言語 MMI システムが作成できる．

ブラウザのロケール設定が日本であれば，図 5 の 1 行目の default.search.label 記述などへ図 7 の

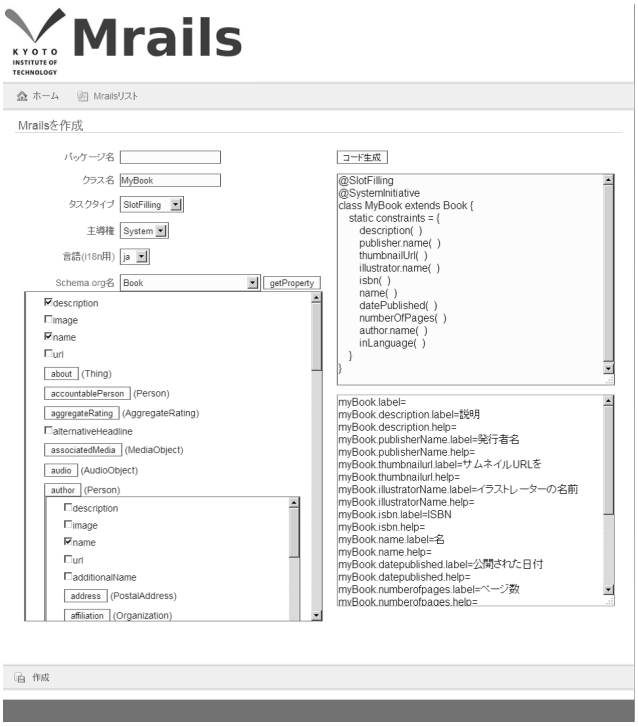

図 8: MrailsBuilder

i18n(internationalization) コードが動的に埋め込まれ，HTML が日本語へローカライズされる．また図 6 の 13 行目の lang 属性にロケール情報を動的に埋め込むことで，対応した音声入出力を行う．

4. MrailsBuilder の作成

MrailsScript を記述するためのサポートを行う環境として，MrailsBuilder を作成した．図 8 では MrailsBuilder によって，スロットフィリングタスクを持った MyBook クラスの MrailsScript の雛形を生成している．

MrailsBuilder では，パッケージ名・クラス名を入力し，タスクタイプ・主導権・言語・Schema.org クラス名を指定する．継承したクラスのプロパティをチェックするか，author.name プロパティのように author インスタンスを辿って name プロパティをチェックすることでデータモデル定義に反映される．検索タスクが指定された場合，検索条件を指定するためのサポートとして，各プロパティの検索条件を選択するドロップダウンリストが表示される．また，選択されたタスクと主導権に必要な i18n コードの雛形の自動生成機能がある．多言語化のサポートとして，日本語の国際化コード記述を選択した言語で翻訳したものも生成される．

5. 評価

MrailsScript ではアノテーションの差し替えのみで違うタスク・主導権のアプリケーション生成できるため保守管理性が高い．MVC モデルを採用したアプリケーションが生成されるためアプリケーション自体の保守管理性も確保されている．

MrailsScript と実装したフレームワークでは GUI をベースにした HTML コードの構造を崩すことなく，HTML 要素を追加する形で MMI に必要な情報を付加しているため，GUI ベースの Web アプリケーション開発方法論がそのまま使用できる．

また対話記述言語から言語依存な情報が全て取り除かれ，インタラクションに必要な言語依存の発話と表示が全て i18n で多言語化されているため，言語ファイルの追加のみで多言語化が行える．

言語レベルでセマンティクスに対応しセマンティック Web への流れに対応するとともに，推論規則の適応が可能であることや，セマンティクスに対応した複数の Web ページの情報をバックグラウンドに持つことができるようになるなどの利点があると考えられる．

6. 今後の展開

今後の展開として，対話管理と発話の意味解釈のための手法の設計が挙げられる．対話管理のためのタスクは，現在は分類した 3 種類から選択するだけであるため，開発者が独自のタスクを簡便に記述する手法を設計する必要がある．また音声認識による混合主導対話にはユーザ発話の意味解釈が必要であり，音声解釈器を持たない Web ブラウザと JavaScript では実装できなかった．意味解釈のために，拡張 HTML に入出力制約として音声認識文法を追加し，ブラウザが発話を解釈してスロットフィリングを行うような MMI ブラウザの実装，または ITSCJ アーキテクチャ第 2 層の認識・合成エンジンのラッパーモジュールに文法を送り，入力モダリティ側で解釈させてスロットフィリングさせるという手法が考えられる．

参考文献

[Heinroth 10] Heinroth, T., Denich, D. Schmitt, A.: OwlSpeak - Adaptive Spoken Dialogue within Intelligent Environments, In Proc. IEEE International Conference pp.666-671 (2010)

[Seneff 98] Seneff, S.et al.: GALAXY-II: A Referenxe Architecture for Conversational System Development. In Proc. ISCSLP 1998, pp.931-034 (1998)

[新田 07] 新田恒雄，桂田浩一，荒木雅弘，西本卓也，甘粕哲郎，川本真一：マルチモーダル対話システムのための階層的アーキテクチャの提案, 情報処理学会研究報告. SLP, 音声言語情報処理, pp.7-12 (2007).

[ITSCJ 10] 情報処理学会情報規格調査会： マルチモーダル対話のための記述言語 Part1 要求仕様 (オンライン). http://www.itscj.ipsj.or.jp/ipsj-ts/ts0012/mmi-arch.html

[W3C 11] Jim Barnett： Multimodal Architecture and Interfaces World Wide Web Consortium (W3C) http://www.w3.org/TR/mmi-arch/