

M-052

ユビキタスパersonalエージェントによるドライブプラン推薦システムの開発 Drive Plan Recommendation System Based on Ubiquitous Personal Agent

岡本 雄三
OKAMOTO Yuzo

長 健太
CHO Kenta

服部 正典
HATTORI Masanori

大須賀 昭彦
OHSUGA Akihiko

1. はじめに

近年、情報技術の発展に伴ってカーナビゲーションシステムが高機能化の一途をたどっており、もはや単なる地図案内だけのシステムでは無くなってきている。HDD や DVD もしくはネットワーク上に、膨大な量の POI(Point Of Interest/施設情報)を格納しており、容易に検索や目的地設定が行えるようになった。しかし、一方でこれらの膨大な情報を基に、複数の POI からドライブプランを作成することは、ユーザにとってはとても煩雑な作業である。既存のナビゲーションシステムにも、ドライブプランの作成支援を行う機能を持つものがあるが、作成されるプランのバリエーションが少なかったり、ユーザの嗜好・状況に合わせたプランを作成するといったきめ細かなパーソナライズを行うことができるシステムは少ない。

一方で、我々はユビキタス環境における状況依存型サービス実現のためのエージェントフレームワークであるユビキタスパersonalエージェント[1]の研究開発を行っている。本フレームワークは、ユビキタス環境におけるさまざまな情報機器から取得したデータからユーザの状況を認識し、その状況に応じたコンテンツ/サービスを選択・実行することで、ユーザに対してパーソナライズされたサービスの実現を行う。状況の認識およびサービスの選択はルールで行っており、ユーザからのフィードバックをそのルールに適切に反映させることで、ユーザの嗜好に追従することが可能である。

そこで、本フレームワークを基にドライブプラン推薦システムの開発を行った。本システムでは、設定された出発・目的地間を仮想的に走行してユーザの状況変化をシミュレートし、その状況や嗜好に適した立寄り地を自動的に追加していくことでプラン作成を行う。また、ユーザが行うプラン作成、修正などの操作をフィードバックとみなし、個々のルールの重要度を自動的に調整することで、ユーザの嗜好へ適応することが可能である。

以下、本稿では第2章でユビキタスパersonalエージェントのフレームワークの詳細について述べ、第3章で本フレームワークによるドライブプラン推薦システムの詳細について述べる。第4章で本システムの評価について述べ、最後に第5章で本稿のまとめを行う。

2. ユビキタスパersonalエージェント

2.1. 概要

本フレームワークは、ユビキタス環境においてユーザの状況を把握し、状況依存型サービスを実現するためのフレームワークである。本フレームワークの特徴は次の通りである。(1)状況の把握、および状況に応じたコンテンツ・サ

ービス選択のためのルールを定義できる。(2)ユーザに対して何故そのコンテンツを選択したかの理由の開示が行える。(3)ユーザに明示的に質問すること、および、ユーザの行動から暗黙的に得るといふ2種類のフィードバック取得の方法がある。以降で、本フレームワークの詳細について述べていく。

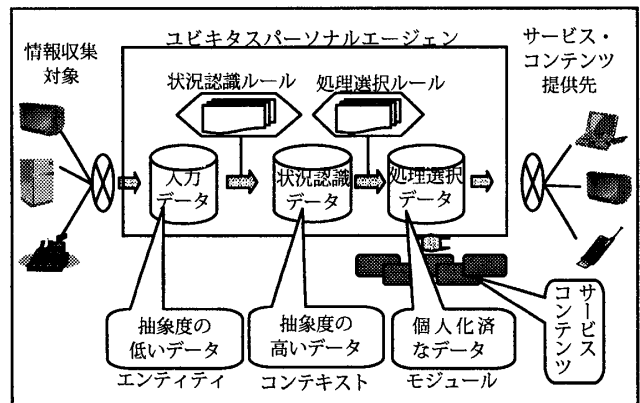


図1. フレームワーク概念図

2.2. アーキテクチャ

本フレームワークの概念図を図1に示す。本フレームワークの流れは次の通りである。1)さまざまな機器から集められた入力データが蓄積される。この入力データをエンティティと呼び、時間や緯度・経度などといった抽象度の低いデータのことを指す。2)エンティティが入力されると、状況認識ルールによって状況認識データが生成され蓄積される。この状況認識データをコンテキストと呼び、ユーザの嗜好や状況などといった抽象度の高いデータのことを指す。3)コンテキストが生成されると、処理選択ルールによって処理選択データが生成される。この処理選択データをモジュールと呼び、個人化済みのデータのことを指す。4)最後に、生成されたモジュールに対応するコンテンツ選択・サービスの実行が行われる。

入力データおよびそれぞれのルールには重みが設定されており、基となったルール、エンティティの重みが加味され、出力データ(コンテキスト、モジュール)にも重みが設定される。生成されたモジュールの重みを考慮して、サービスの実行の可否を判断することで、パーソナライズされたサービスの質をコントロールできる。また、ルールの重みを後で述べるフィードバックの機構によりユーザ毎に調整することで、それぞれのユーザの行動パターンに適したサービスを実現することが可能である。

2.3. ルール

本フレームワークのルール記述例を図2に示す。この例は、車の現在位置(Coordinate)と施設情報(POI)から近隣施設情報(NearbyPOI)を生成する状況認識ルールの例である。

†株式会社東芝 研究開発センター
Corporate Research & Development Center: TOSHIBA Corporation

```
rule DerivateNearbyPOI {
  //重みの定義
  weight = 1.0;

  //出力と入力の定義
  NearbyPOI nearbyPoi(POI poi, Coordinate coordinate) {
    dist = distance(poi.lat, poi.lon, coordinate.lat, coordinate.lon);

    if(dist < 1000) {
      nearbyPoi.id      = poi.id;
      nearbyPoi.name   = poi.name;
      nearbyPoi.lat    = poi.lat;
      nearbyPoi.lon    = poi.lon;
      . . . 中略 . . .
      nearbyPoi.dist   = dist;
    }
  }
}
```

図2. ルール記述例

また、適切なパーソナライズを行うためには、粒度の細かいルールを沢山定義する必要がある。このため、ルールによってはルール中の一部の値のみ(例えば比較式中の対象)が異なり、それ以外の部分はまったく同じ記述になるものがある。このような同様なルールを複数記述することを防ぐために、列挙型というものを用意して1つのルールとして記述できるようにしている。列挙型を利用したルールの例を図3に示す。この例は、近隣施設情報(NearbyPOI)とユーザの食事の好み(FoodPreference)からお勧めの施設(RecommendPOI)を生成する処理選択ルールの例である。ルールの冒頭で列挙型の宣言が行われていて、名前(ex. FoodGenre)と値(ex. 和食,洋食,中華)の定義を行う。続いて、列挙型の各値の組合せにおけるルールの重みの定義を行う。ルール内での列挙型の変数名(ex. foodGenre)を定義し、次に重みの定義を行っている。フレームワーク内部では、ルール内の列挙型の変数名が、実際の列挙型の値に置換され、この場合だと3×5=15のルールに展開される。展開後の各ルールの重みは、図内の表に示す通りに展開される。

2. 4. フィードバック

ユーザに対し適切なサービスを実行するために、個々のルールに対してフィードバックを行い、その重みの調整を行っている。フィードバックの方法として、明示的および暗黙的の2種類の方法が用意されている。

各出力データ(コンテキスト, モジュール)には、それを出力したルールが紐付けされており、そのコンテンツを選択した理由や、その状況を認識した理由をユーザに提示することが出来る。そこで、そのコンテンツ選択が適切であったか、その状況認識が適切であったかをユーザに明示的に質問して、それらを出力したルールに対してフィードバックを行うのが明示的フィードバックである。

暗黙的フィードバックでは、コンテンツ配送後にユーザがどのような行動を行ったかを入力データとして受け取り、その入力データをフィードバックとする方法である。暗黙的フィードバックでは、フィードバックの対象となるルール内で定義を行うが、図3のルール例ではその定義がなされている。この例では、ユーザに対してあるジャンルの食べ物屋を推薦したときに、その後ユーザが実際にそのお店を訪れたというデータ(VisitingPoint)を受け取ったら、そのジャンルのお店を勧めたルールの重みを2倍にするといった定義がなされている。

```
rule DerivateRecommendPOI {
  //列挙型の定義
  FoodGenre = ["和食" "洋食" "中華"];
  POIClass = ["寿司" "そば" "カレー" "スパゲティ" "ラーメン"];

  //列挙型の各値のときにおけるルールの重みの定義
  weights[FoodGenre foodGenre, POIClass poiClass] {
    [* *] = 0.1;
    ["和食" "寿司", "そば"] = 1.0;
    ["洋食" "スパゲティ"] = 1.0;
    ["中華" "ラーメン"] = 1.0;
    ["和食" "カレー", "ラーメン"] = 0.5;
  }

  RecommendPOI recommendPoi(NearbyPOI nearbyPoi,
    FoodPreference foodPref) {
    if(nearbyPoi.class == poiClass && foodPref.type == foodGenre)
    {
      recommendPoi.id = nearbyPoi.id;
      recommendPoi.lat = nearbyPoi.lat;
      recommendPoi.lon = nearbyPoi.lon;
      . . . 中略 . . .
      recommendPoi.dist = nearbyPoi.dist;
    }
  }

  //暗黙的フィードバックの定義
  feedback(VisitingPoint visitingPoint) {
    case(recommendPoi.name == visitingPoint.name) {
      $weight = $weight * 2.0;
    }
  }
}
```

この変数が列挙型の各値に置換される

展開後の 重み付け		poiClass				
		寿司	そば	カレー	スパゲティ	ラーメン
FoodGenre	和食	1.0	1.0	0.5	0.1	0.5
	洋食	0.1	0.1	0.1	1.0	0.1
	中華	0.1	0.1	0.1	0.1	1.0

図3. 列挙型とフィードバック定義のルール記述例

3. ドライブプラン推薦システム

3. 1. システム概要

まず始めに、本システムを使ってドライブプランを作成し、模倣的に走行を行うまでの一連の流れについて述べていく。システム全体は、PC上でVGAサイズのUIとして表示され、タッチパネル付のカーナビシステムの画面を模倣して作成を行った(図4, 5)。

始めに、ドライバや同乗者の嗜好や、ドライブの目的などを設定するユーザカードが表示される。ユーザカードでは、目的・同乗者・食事好み・予算・性別・年代といったものが設定できる。設定されたユーザカードにしたがって、システムが適切なPOIの推薦を行うことになる。次に出発・目的地設定画面に移り、プラン生成の際の出発・目的地および出発・到着時間の設定を行う。

出発・目的地の設定が終わると、システムが自動的にユーザにとって適切なPOIを、到着時間を加味して選択し、複数のドライブプランを作成する(図4)。プランに追加されたPOIのボタンを押すことで、POI詳細画面に移動する。また、決定ボタンを押すことで、現在表示されているプランが選択され、シミュレート走行画面に移動する。

POI詳細画面では、選択したPOIの説明文と到着・出発時間が表示される(図5)。また、画面上部には本システムの特徴であるPOI選択の理由をユーザに示すコンテキストが表示される。その理由が不適切な場合は、ボタンを

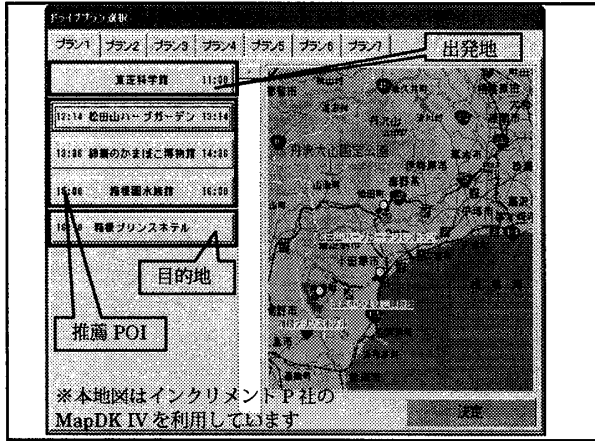


図4. ドライブプラン選択画面

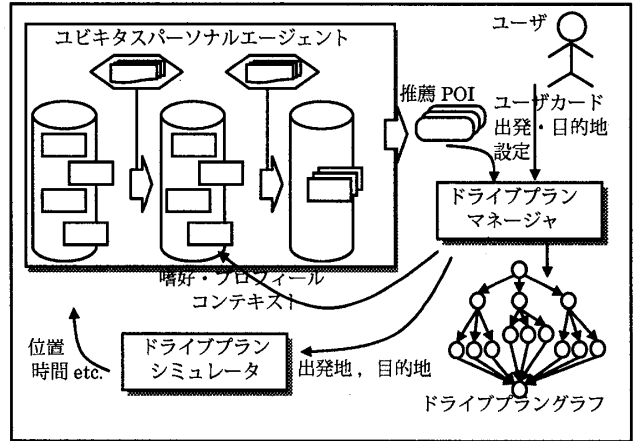


図6. システムアーキテクチャ

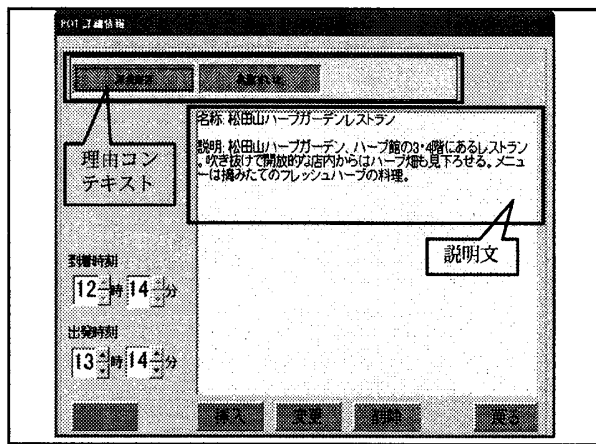


図5. POI詳細画面

押すことで、そのコンテキストをキャンセルすることができる。また、削除ボタンでその POI をプランから削除でき、変更・挿入ボタンで POI の変更・挿入が行える。変更・挿入の場合、システムが推薦する代替候補地の他に、ユーザー自身がジャンルおよび地図から POI を選択することが可能である。

シミュレータ画面は、選択されたドライブプランに従ったドライブを模擬するための画面である。この画面にあるコンテキストボタンを押すと、システムが推定するユーザーの現在のコンテキストが表示される。コンテキストは設定およびキャンセルが可能で、突発的な状況変化によるプランの変更が行える。例えば「トイレに行きたい」というコンテキストを設定することで、システムはトイレのありそうな近隣の POI を自動的に設定する。

3. 2. アーキテクチャ

本システムのアーキテクチャを図6に示す。本システムでは、前章で述べたユビキタスパーソナルエージェント本体と、ドライブプランを作成する際の候補 POI の管理・プラン作成を行うドライブプランマネージャ、およびユーザーの走行を模擬する目的で利用するドライブプランシミュレータから構成される。

プラン作成の流れは次の通りである。まず、ドライブプランマネージャが、設定されたユーザーカードに基づき

エージェントへコンテキストの設定を行う。次に、設定された出発・目的地間のルートの検索を行い、そのルートをドライブシミュレータで仮想的に走行し、その時の位置情報と時刻をエンティティとしてエージェントへと供給する。エージェントでは供給されたエンティティによりルールが評価され、コンテキストおよびモジュールが生成される。ドライブプランマネージャが生成された複数の推薦 POI をドライブプラン生成のためのグラフへ挿入する。次に、生成された POI から目的地間のルート検索を行って、仮想走行を行う。以上を繰り返す、ドライブプラングラフの作成を行う。

作成されたグラフからドライブプランを作成する方法は次の通りである。作成されたグラフの全てのパスについて、POI の重みを合計して、その値が最大となるものを選択し、それをプラン候補とする。次に、そのプラン候補に含まれる各 POI の重みを減少させる。これは、複数のプランに同一の POI が含まれるのを防ぎ、プランにバリエーションを持たせるためである。以上の動作を、規定数のプランが作成されるまで繰り返すことで、ドライブプランの作成を行う。

3. 3. エンティティ・コンテキスト・モジュール

本システムで利用したエンティティ、コンテキスト、モジュールを表1にまとめる。

エンティティには大きく分けて、動的・静的なエンティティがある。静的なエンティティはシステムの初期化時に代入され以降は変わらない。コンテキストには大きく分けて、短・中・長期的状態、嗜好・プロフィール、ユーザーの行動をあらわすコンテキストがある。嗜好・プロフィールのコンテキストは、ユーザーカードに対応したものである。行動をあらわすコンテキストは後ほど述べる暗黙的フィードバックとして利用している。

3. 4. ルール

次に、本システムで利用した状況認識、処理選択ルールについて述べていく。

状況認識ルールには、POI・閾値・現在位置から近隣 POI を、現在日時から時間帯を、現在日時から空腹状態を、現在日時から季節を生成するルールがある。これらのルールから生成されるコンテキスト以外は、全て外部から

表1. エンティティ・コンテキスト・モジュール定義

エンティティ	
動的	現在位置(Coordinate), 現在日時(Date) 近くを表す閾値(NearbyThresholod) 立寄り施設(VisitingPoint)
静的	施設情報(POI)
コンテキスト	
短期的状態	近隣施設(NearbyPOI)
中期的状態	時間帯(Time), 空腹(Hungry) のどが渇いた(Thirsty), 眠い(Sleepy) トイレに行きたい(Urinary)
長期的状態	季節(TimeOfYear)
嗜好・プロフィール	性別(UserGender), 世代(UserGeneration) 同乗者(Passenger), 目的(Purpose) 予算(Budget), 食事の好み(FoodPreference) アウトドア・インドア志向(ActivityType)
行動	検索(SearchedPOI) 施設立ち寄り(VisitingPoint)
モジュール	
	推薦 POI(RecommendPOI)

直接設定される。

処理選択ルールについては、すべて推薦 POI の生成を行うルールである。ユーザの現在の中・長期的状態もしくは嗜好・プロフィールに適切と思われるジャンルと、近隣施設のジャンルとが一致するなら、その施設を推薦 POI として生成するという定義がなされている。近隣施設のジャンル(約 300 種類)および状態・嗜好の属性の値は、図 3 のルールの例のように列挙型で定義しており、ある状態・嗜好に適切と思われるジャンルを推薦するルールの重みをあらかじめ高く設定しているため、始めからある程度適切なドライブプランの作成が行える。更に、次に述べるフィードバックにより、システムを利用していくにつれてルールの重み付けがユーザの嗜好に合わせて調整され、より適切なプランの作成が行えるようになる。

3. 5. フィードバック

本システムでは、明示・暗黙的フィードバック両方を利用してルールの重み付けの調整を行っている。

明示的な方法として、提示された POI の削除および変更、POI を導いた理由であるコンテキストのキャンセルがある。POI の削除では、その POI を導いたルールに対して負のフィードバックがかかる。POI の変更では、まず変更前の POI に対して削除と同様のフィードバックをし、変更後の POI が代替候補から選択された場合は、その POI を導いたルールに正のフィードバックをかける。コンテキストのキャンセルの場合は、そのコンテキストを導いたルールに対して負のフィードバックをかけている。

暗黙的な方法については、ユーザの行動を表すエンティティをフィードバックとみなすが、本システムでは、施設立ち寄り、および検索行為といった 2 つのエンティティを利用している。プラン作成後にユーザがどの POI に立寄ったかを表すエンティティが入力されると、その POI と同じジャンルの POI を勧めたルールに正のフィードバックがかかる。同様に、ユーザがある POI を検索しプランに追加した

際に、その行動を表すエンティティが入力され、同じジャンルの POI を勧めたルールに正のフィードバックをかけている。

4. 評価

ユーザの継続的な利用を通して、本システムがどの程度ユーザの嗜好に適合していくかを、仮想ユーザモデルを利用して評価を行った。ユーザモデルでは、あらかじめ定義してあるユーザの嗜好に基づき、システムから提示される複数のプランをスコア付けし、最高得点のプランを選択、プランに含まれる低スコアの POI をよりスコアの高い POI に変更を行うといった実ユーザを模擬する動作を、選択したプランのスコアがそれ以上増えなくなるまで繰り返す。同一のユーザモデル、同一の目的・出発地間について、プラン作成・模擬走行を 15 回繰り返し行った結果を図 7 に示す。

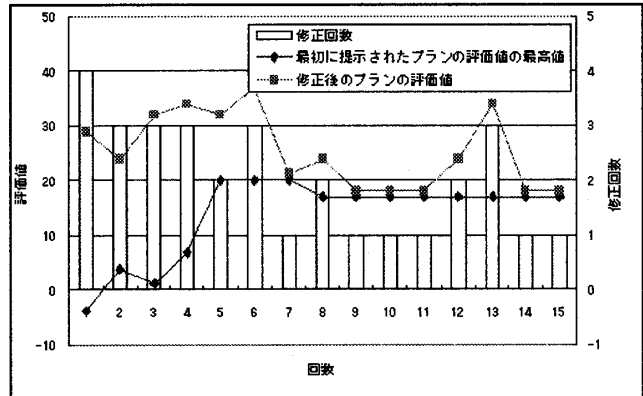


図7. プラン修正前・後の評価値と修正回数

結果から、繰り返し回数が進むにつれシステムから提示されるプランの評価値の最高値が上がり、ユーザによるプランの修正回数が減っていくことが分かる。つまり、継続的な利用を通して、システムがユーザの嗜好に適合していることが確認できる。

5. まとめ

本論文では、ユビキタス環境向けエージェントフレームワークであるユビキタスパーソナルエージェントによるドライブプラン推薦システムについて述べてきた。本システムは、ユーザの嗜好や状況を把握し、適切なドライブプランを自動的に作成する。ユーザに対して何故その POI を選択したかの理由を提示でき、ユーザからの明示的・暗黙的なフィードバックを受け取って、ユーザの嗜好に合わせて自動的に提案する POI の調整を行えることを評価によって確認できた。

今後の課題としては、現状はエンティティ・コンテキストとして容易に取得できるものしか利用していないが、車載センサーなどとの連携によって、より高度な状況に応じた POI の推薦を行うといったことがあげられる。

参考文献

- [1] 山崎智弘, 長健太, 服部正典, 大須賀昭彦: ユビキタスパーソナルエージェントシステムの実装と評価. 第 66 回情報処理学会全国大会(2004).