ユビキタス環境向け知的移動エージェント picoPlangent

長 健太　　林 久志　　大須賀 昭彦
株式会社 東芝 研究開発センター コンピュータ・ネットワークラボラトリー

1 まえがき

近年，PDA や携帯電話などを使うことで，あらゆる場所，あらゆる場面で情報を扱うことが可能となるユビキタス環境を実現することが可能になった．しかし，それらの携帯機器は非常に少ないリソースしか持たず，また画面サイズの大きさなど，ユーザインタフェースに制約があるため，ユーザは多くの情報を一度に扱うことはできない．携帯機器上で情報を扱うシステムでは，情報はユーザに合わせて適切に要約され，適切なタイミングで提示される必要がある．

携帯機器上で動作する移動エージェント [2][3][4][5] を用いることで，ユーザ向けにパーソナライズされた情報提供システムを構築することができる．エージェントは携帯機器上でユーザの動作を観察し，ユーザの興味を学習することで，ユーザが必要とする情報を必要な時に提示することができる．

多くの携帯機器は，そのリソースが非常に限られているため，携帯機器上で動作するエージェントはコンパクトである必要がある．また，ユーザに適切な情報提供が行なえるよう，エージェントが知的である必要がある．

本稿では，我々が開発したユビキタス環境向け知的移動エージェント picoPlangent について説明する．

2 アーキテクチャ

picoPlangent はユビキタス環境で動作する知的移動エージェントである．ユーザが利用するさまざまなデバイスを跨って動作することを可能にするため，以下の特長を備えている．

- 小さいフットプリント

  リソースの制限が厳しい携帯機器上でエージェントを動作させるには，エージェントのサイズを極めて小さくする必要がある．picoPlangent では，エージェントの各機能は小さなコンポーネントに分割され，コンポーネントは携帯機器に分散配置される．エージェントの機能がコンポーネントとして抽出されているため，それらコンポーネントを利用して動作するエージェント本体を小さくすることができる．

- ゴールベース

  エージェントはユーザの要求を表すゴールを保持し，ゴールを解決すべく動作する．ゴールは汎用的な形式で表現され，さまざまなデバイス上で容易に扱えるようになっている．

- プランニング

  picoPlangent は Plangent[1] に基づくプランニング機能を持つ．エージェントがゴールを解決できない場合，プランニングによる代替プランの生成を行なうことができ，エージェントの柔軟な動作を可能にしている．

- 移動

  エージェントはさまざまなデバイス間を，さまざまな通信手段を用いて移動することができる．

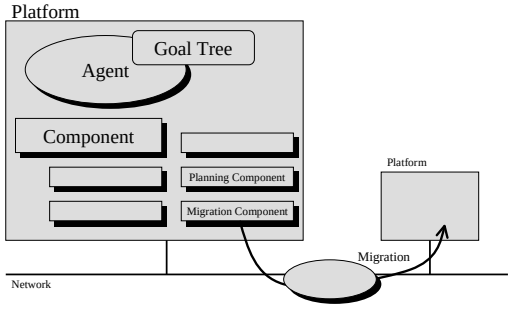

図 1: ソフトウェアアーキテクチャ

2.1 エージェントおよびゴール

picoPlangent は，エージェント，コンポーネント，プラットフォームから構成される（図 1）．

An Intelligent Mobile Agent System in Ubiquitous Environment: picoPlangent
Kenta Cho, Hisashi Hayashi, Akihiko Ohsuga
Computer & Network Systems Laboratory, Corporate Research & Development Center, Toshiba Corporation

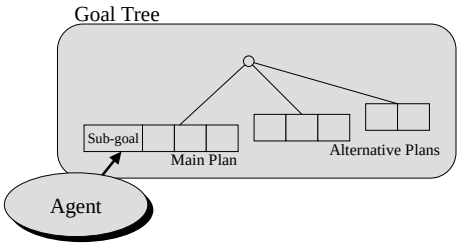

図 2: ゴールツリー

picoPlangent のエージェントは，ゴールと呼ばれる，Prolog の項の形式で表記したユーザの要求を解決すべく動作する．エージェントは，ユーザからゴールを受けとった後，ゴールを解決するプランを生成する．プランはいくつかのサブゴールの並びから構成され，エージェントはそれらサブゴールを順に解決することで，ユーザの要求を解決する．

エージェントは，ユーザから与えられたゴールを解決する，複数のプランを持つことができる．それらのプランは，1 つのメインプランといくつか代替プランから構成され，ゴールツリーと呼ばれる構造で管理される（図 2）．

ゴールを解決するために，エージェントは以下のように動作する．

1. メインプラン中の最初のサブゴールを読み込む．
2. コンポーネントを利用してサブゴールを解決する．
3. エージェントがサブゴールの解決に成功した場合，そのサブゴールはメインプランから削除される．代替プランの先頭のサブゴールが，そのサブゴールと同じであった場合は，そのサブゴールも削除される．
4. エージェントがサブゴールの解決に失敗した場合，メインプランはゴールツリーから削除され，代替プランの 1 つがメインプランとなる．
5. これらの動作をメインプラン中のすべてのサブゴールが解決されるか，すべてのプランが削除されるまで行う．

すべてのサブゴールが解決された場合，エージェントはゴールを解決できたこととなる．そうでない場合，エージェントはゴールの解決に失敗したこととなる．

2.1.1 コンポーネント

コンポーネントはサブゴールをエージェントから受けとり，サブゴールを解決するための動作を行なう．コンポーネントは，開発者が各アプリケーション機能に合わせて実装するものと，エージェントの移動や，プランニング機能など，picoPlangent の基本機能として提供されるものがある．

```
action(print(Msg)).
action(goto(_)).
axiom(message([],_),[]).
axiom(message([Node|Rest],Msg),
      [goto(Node),print(Msg),
       message(Rest,Msg)]).
```

図 3: プラナーのルールサンプル

- 移動コンポーネント

  移動コンポーネントは，エージェントを機器間で移動させるためのコンポーネントである．エージェントは，移動コンポーネントでシリアライズされ，ネットワーク接続された他の機器へ移動する．移動コンポーネントには，エージェントを送信するプロトコルに対応したいくつかの種類がある（RmiMirgration, HotsyncMigration, SOAPMigration）．

- プランニングコンポーネント

  プランニングコンポーネントは，ゴールからゴールツリーを生成する．ゴールツリーは，ゴールをサブゴールに分解する方法を示したルールに従って生成される．

  ルールは Prolog の項の形式で表記される（図 3）．ルールはアクション宣言（action）および分解ルール（axiom）から構成される．アクションはコンポーネントによって解決可能なサブゴールであり，プラナーはアクションをこれ以上分解しない．分解ルールは，アクション以外のサブゴールを複数のサブゴールに分解するルールである．

2.1.2 プラットフォーム

プラットフォームは，各機器上のエージェントとコンポーネントを管理するための場所である．コンポーネントはプラットフォームに登録され，エージェントからサブゴールの解決が依頼された場合に，プラットフォームはそのサブゴールを解決する適切なコンポーネントを提示する．プラットフォームは以下の手順でコンポーネントを検索する．

1. エージェントがサブゴールをプラットフォームに送信する．

2. プラットフォームが登録されている各コンポーネントにそのサブゴールが解決可能かを質問する．
3. プラットフォームが，解決可能と答えたコンポーネントのリストを生成し，エージェントに送信する．
4. エージェントはリストからコンポーネントを 1 つ選択し，そのコンポーネントにサブゴールの解決を依頼する．

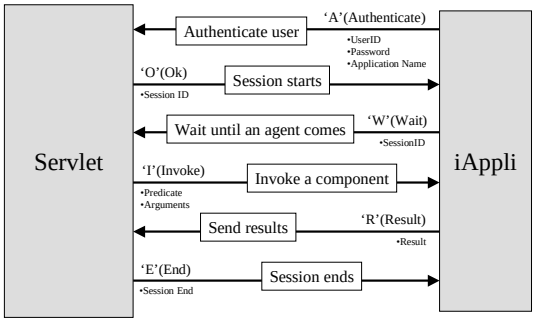

図 4: iAppli プラットフォームの通信プロトコル

3 実装

3.1 PC 上での実装

PC 上の picoPlangent は，J2SE で実装されている．PC 上のエージェントは Agent クラスで実装される．開発者は，Agent クラスを拡張することで，エージェントのライフサイクルを変更することができる．

コンポーネントは，Component インタフェースを実装するクラスで実現される．Component インタフェースは，あるゴールがそのコンポーネントで解決可能かどうかを返すメソッドおよびエージェントがコンポーネントにゴールの解決を依頼する際に呼び出されるメソッドを備える．

プラットフォームは，Platform インタフェースを実装するクラスで実現される．基本的な機能を備えたプラットフォームは，DefaultPlatform クラスで実現されている．

他のプラットフォームからエージェントの移動を受け付ける場合は，エージェントを生成するためのユーティリティクラスを利用する．例えば，Java RMI によるエージェントの移動を受け付ける場合，RmiAgentCreator クラスを用いて，RMIRegistry へのプラットフォームの登録を行なう．エージェントは，移動コンポーネント（RMIMigration）によってシリアライズされ，送信される．

3.2 Palm 上の実装

Palm デバイス上では，picoPlangent は GCC で実装されている．プラットフォームおよびコンポーネントは PalmOS のアプリケーションとして実装されている．エージェントを PC から Palm に移動させる通信路としては，Palm と PC のデータを同期させる機構である Hotsync を利用している．

エージェントが移動する際には，HotsyncMigration コンポーネントが利用される．HotsyncMigration は以下の手順でエージェントを移動させる．

1. ゴールツリーおよびエージェントの保持するデータを PalmOS 上のプラットフォームで解釈可能なバイト列にシリアライズし，ファイルに保存する．
2. Hotsync が起動されると，ファイルに保存されたデータがコンジットによって PalmOS のデータベースに書き込まれる．
3. PalmOS 上のプラットフォームがデータベースを読み取り，ゴールツリーおよびエージェントの保持するデータを復元する．

3.3 iAppli 上の実装

iAppli は NTT Docomo の携帯電話上で動作する Java VM である．iAppli は J2ME をベースとしているが，プログラムサイズの制限が非常に厳しい．クラスファイルは，JAR(Java Archive) で圧縮した状態で 10 キロバイトを越えてはならない．

この制限内でエージェントを動かすため，iAppli 上の picoPlangent はプロキシコンポーネントと呼ばれるコンポーネントを利用する．プロキシコンポーネントは，サーブレット上で動作し，エージェントから送られるゴール解決要求を HTTP で iAppli 上のコンポーネントへ送信する．

iAppli 上の picoPlangent は 2 つのクラスから構成される．

- ServerConnector

  ServerConnector は，サーブレット上のプロキシコンポーネントから送信される要求を受信し，iAppli 上のコンポーネントへ処理を依頼，結果をプロキシコンポーネントに送信する（図 4）．

- Component class

  PC 上の picoPlangent では，1 つのコンポーネントは 1 つのクラスで構成されている．iAppli 上では，プログラムサイズの削減のため，コンポーネント

は 1 つのクラス内に実装されている．開発者はコンポーネントのプロパティファイルを作成し，このファイルから生成されたコンポーネントクラスのスケルトンを実装する．

4 評価と今後の課題

picoPlangnt は，リソースが非常に限られたデバイス上で動作する知的移動エージェントである．J2ME をベースとした JavaVM は多数あるが，iAppli は格納可能な JAR ファイルのサイズが 10 キロバイトという，特に厳しいリソースの制限がある．

picoPlangent エージェントは，非常にシンプルなライフサイクルを持ち，各機能がコンポーネントとしてプラットフォームから分離されている．そのため，picoPlangent のプラットフォームは非常にコンパクトにすることができる．iAppli 上のプラットフォームのサイズは約 4.6 キロバイト，Palm 上のプラットフォームのサイズは約 6 キロバイトである．

エージェントのライフサイクルはゴールツリーで表現され，携帯機器上での柔軟な動作を実現する．ゴールツリーの主な目的は，プラナーの持つ機能を，プランニングコンポーネントを配置可能なリソースを持たない携帯機器上で利用することである．ゴールツリーはプランニングコンポーネントで作成されたメインプランと代替プランを管理する．エージェントはゴールツリーを持ち歩き，あるデバイス上でメインプランの実行に失敗した場合，あらかじめ生成してある代替プランを利用することができる．

iAppli 上のプラットフォームでは，サーバ上のプロキシコンポーネントを利用することで，プラットフォームサイズを削減している．しかし，プロキシコンポーネントを利用することで，サーバと携帯電話間の通信頻度が増すという問題がある．

我々は携帯電話上でエージェントのライフサイクルを扱うプラットフォームも作成した．ターゲットデバイスとしては，iAppli よりも格納可能なクラスサイズが増加している，J-Phone および au の携帯電話を想定している．これらの実装では，エージェントのライフサイクル管理部が携帯電話側に実装され，通信量の削減をはかっている．エージェントをシリアライズするためのシリアライザおよびデシリアライザがサーブレットおよび携帯電話上に配置される．

携帯電話上のコンポーネントが利用できるプログラムサイズは限られるため，開発者は携帯電話上に配置するコンポーネントを適切に設計する必要がある．いくつかのコンポーネントはサーバ上に配置され，エージェントが必要に応じてサーバに一時的に戻り利用する．

我々は，エージェントにコンポーネントの自動再配置機能を備えることを検討している．エージェントは，各コンポーネントの利用頻度を記録し，より頻繁に使われるコンポーネントを携帯電話上に配置することで，サーバと携帯電話間の通信を減らすことができる．

5 むすび

picoPlangent のアーキテクチャを述べ，アーキテクチャを PC，Palm および iAppli で実装する方式について述べた．エージェントはさまざまなデバイス上をさまざまなプロトコルを利用して移動することができる．picoPlangent のプラットフォームは小さいフットプリントで実装することができ，実装方式をデバイスのリソース制限に合わせて変更することも容易である．ゴールツリーはエージェントのライフサイクルを簡潔にするとともに，携帯機器上でのエージェントの柔軟な動作を実現する．

参考文献

[1] Ohsuga,A., Nagai.Y., Irie,Y., Hattori,M., and Honiden,S. :PLANGENT: An Approach to Making Mobile Agents Intelligent, IEEE Internet Computing, Vol.1, No.4(1997), pp.50-57; http://computer.org/internet/ic1997/w4050abs.htm

[2] Ciminiera,L., Maggi,P., Sisto,R.: SCRAB: innovative services supporting user and terminal mobility, IEEE Distributed Computing System Workshop, 2001 International Conference on, 2001, pp.487-493.

[3] Bergenti,F., Poggi,A.: LEAP: A FIPA platform for handheld and mobile devices, International Workshop on Agent Theories, Architectures, and Languages (ATAL-2001).

[4] Mahmoud,Q.: MobiAgent: A mobile agent-based approach to wireless information systems, Agent-Oriented Information Systems (AOIS-2001) 2001, p. 87-90.

[5] OMRON SOFTWARE: Jumon. http://www.jumon-agent.com/.