MobileAgentFrameWork“Café”の提案

2E－5　清水 章央　田中 卓弥　濱井 龍明　井ノ上 直己
(株)KDDI研究所

1. はじめに

近年 Web 上のサービスは、膨大な量のサービスが多種多様な形で提供されているが、ユーザは自身の目的に応じてサービスを選択するものの、目的にあったサービスを探すのは難しい上、その目的を達成する為には、いくつものサービスサイトにアクセスしなければならない。我々はそれらの負担を解消する為、Web サービスを効果的に連携統合させユーザに提供する事を考え、ネットワーク上のコンピュータ間を移動しながら処理を行うプログラムであるモバイルエージェントを利用したフレームワークを開発している。モバイルエージェント技術を利用したシステムは、AgentSpace[1]、Voyager[2]、Aglets[3]があるが、これらはひとつのアプリケーション毎にモバイルエージェントを開発しなければならなく、またユーザ側でもいろんなモバイルエージェントが存在し、管理が煩雑になる。Café ではモバイルエージェントの汎用化を目指して設計してあり、移動先でアプリケーションに必要なコードを生成でき、管理するモバイルエージェントは 1 つでよい。本稿では本 MobileAgentFrameWork”Café“の構成および機能について述べる。

2. Café フレームワーク

2. 1コンセプト

MobileAgentFrameWork”Café”は以下のコンセプトのもと設計した。

1. サービスを提供する側は、自由にモバイルエージェントサービスをユーザへ提供できる。
2. ユーザはサービスカテゴリの違いを意識することなく、リクエストに応じたサービスを利用できる。

これらのコンセプトは、サービスエージェントのネーミングトポロジ自動登録とモバイルエージェントのアクションシナリオ選択によるエージェントダイナミックコード生成により実現される。

Proposal of Mobile Agent Frame Work “café”
Akihisa Shimizu , Takaya Tanaka , Tatsuaki Hamai , Naomi Inoue
Internet Application Laboratory
KDDI R&D Laboratories Inc.

2. 2フレームワーク構成

モバイルエージェントフレームワークは、モバイルエージェント、コントロールエージェント、サービスエージェント(スタイルエージェントはサービスエージェントの１種)、エージェント実行環境(AgentServer)からなる。

Café フレームワークは図 2.2①に示すような構成をとっている。またエージェントスタックを図 2.2②に示す。

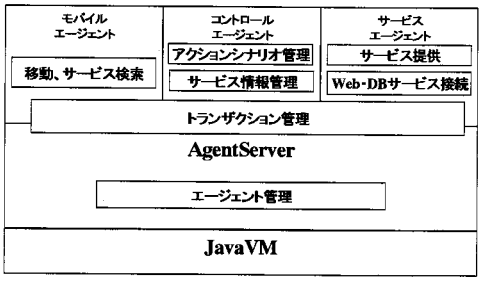
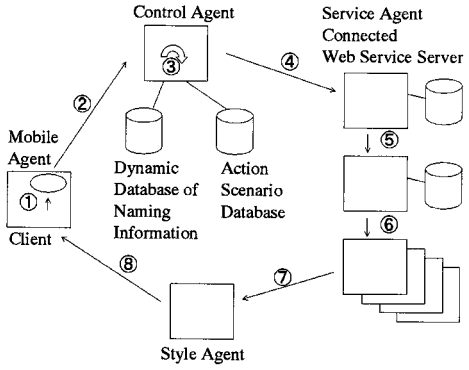


図 2.2①エージェントスタック



①リクエスト受付
②Control Agent接続
（Naming InformationとAction Scenarioの取得）
③・④・⑤・⑥Stationary Agent接続
（Webサービスのコンテンツ取得）
⑦Style Agent接続
（集めたコンテンツの整形）
⑧Userへ

図 2.2②Café フレームワーク構成

フレームワークはモバイルエージェント、コントロールエージェント、サービスエージェント、スタイルエージェントで構成される。これらのエージェントはエージェントマネージャー上で実行され、各マシンの

エージェントにより管理される。また各エージェントは各々でトランザクション管理を行う。 クライアントからのリクエストを受付けると、(①)クライアント端末で待機中の MobileAgent はまずリクエストを持って、ControlAgent にアクセスする。(②)Control Agent では、ActionScenarioDatabase からリクエストにマッチした ActionScenario を持ち出し、必要な Service Agent にアクセスするようにコードを自動生成する。(③)MobileAgent は ActionScenario より生成された処理を、各 ServiceAgent に移動しながら結果を収集する。(④～⑥)収集したデータを ActionScenario に付属の CascadingStyleSheets と一緒に StyleAgent に渡すことにより、html として結果が返される。(⑦)生成された html データをクライアントに持ち帰り表示する。(⑧)

3. 各エージェントと移動処理について

3. 1モバイルエージェント

モバイルエージェントは、リクエストに対応したコントロールエージェントから指定されたアクションシナリオに基づいて行動し、ユーザへと結果を持ち帰る。このエージェントの振る舞いを決定するアクションシナリオは、以下のような構成になっている。

```
<?xml version="1.0"?>
<servicename search1="xxx" search2="xxx">xxxxxx
  <host addr="xxx.xxx.xxx.xxx">
    <method arg="xxx.xxx.xxx.xxx:xxx">xxxx</method>
    <sync type="x" group="x" />
    <order>x</order>
    <case>xxxx</case>
  </host>
  <host addr="xxx.xxx.xxx.xxx">
    <method arg="xxx">xxxxxxx</method>
    <sync type="x" group="x" />
    <order>x</order>
    <case>xxxx</case>
  </host>
  <allofcase>xxxx</allofcase>
  <cssfile>xxxx</cssfile>
</servicename>
```

アクションシナリオ本体は、エージェントの振る舞いが XML で記述されている。ルート要素として servicename タグがあり、どのようなサービスか、またユーザのどんな要求に答えうるサービスかが、属性に記述してある。host タグはサービスを抽出してくるサービスエージェントのアドレスが記述されてあり、その中は

method タグ・・・どんなメソッドを使えばよいか
sync タグ・・・トランザクション処理の必要性
order タグ・・・順序の必要性
case タグ・・・サービスの例外処理

がひとまとめとして host タグに関連付いている。また method タグには属性として argument の指定ができ、host+method を指定するとそのサービスを利用して収集したデータを用いて、次のサービスを受けることができる。

allofcase タグは1つの連携サービスとしての例外処理を記述することができる。cssfile タグは css ファイルの名前が記述してある。

3. 2サービス、コントロールエージェント

コントロールエージェントは、各サービスエージェントのアドレスとサービス種別を管理している。この管理情報は、サービスエージェントが立ち上がったときにネットワーク的に近いコントロールエージェントにサービスエージェントが自律的に登録する。これによりネットワーク的に近いところで小さなトポロジが出来上がり、サービス管理の負荷分散が計れる。全体的なトポロジは図3. 2のようになっている。

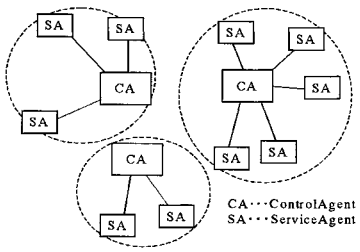


図3. 2ネームサービストポロジ

4. アプリケーションモデル

例えば出張案内のアプリケーションを Café を使って考えてみる。モバイルエージェントは時刻表の検索、新幹線や飛行機の予約、ホテルの予約、などをそれぞれのサービスエージェントへアクセスし予約を完了してくれる。この時、sync タグによって、それぞれの予約が絶対必要かオプションかの選択も可能であるし、満員のときなども case タグによって違うホテルを探すアクションシナリオを探し予約を完了したり、新幹線の時間を変更して予約を行う例外処理の対応などもできる。

5. おわりに

Mobile Agent Frame Work「Café」による WEB サービスの効果的な連携統合について述べた。

参考文献

[1]AgentSpace http://islab.is.ocha.ac.jp/agent/index.html
[2]Voyager http://www.objectspace.com/products/voyager/
[3]Aglets http://www.trl.ibm.com/aglets/index-j.html