

Innovations by InterSystems

Caché における POJO サポート
InterSystems Jalapeño Technology

技術白書

Andreas Dieckow
Principal Product Manager, Strategic Planning
InterSystems Corporation

はじめに

Java プログラミング言語は「一度書けば、どこでもデプロイ可能」な特徴を持つが故に、多くの熱烈な信奉者や開発者がいます。しかし、Java はオブジェクト指向言語であり、そのデータオブジェクトは本質的に、永続性を持ちません。Java の熱烈な信奉者でさえ、リレーショナルデータベースで Java オブジェクトに永続性をもたせるために行なう、オブジェクトとリレーショナルテーブル間のマッピング(O/R マッピング)は単純作業でしかない、と認めます。ある推定によれば、開発労力の 60% から 70%が、マッピングに費やされているといわれます。

インターシステムズ社の Caché は、リレーショナルデータベースのように JDBC をサポートするだけでなく、Java アプリケーションにデータの永続性を与えるための、オブジェクト指向の手法を提供しています。Caché クラスは Java のプロキシクラス、または 高パフォーマンスの Bean-Managed パーススタンスを備えた Enterprise Java Beans に投影することができます。このアプローチを行なった場合、O/R マッピングは不要ですが、Java 開発者は、Java の開発環境で投影し、それを使用するためには、Caché 内でデータオブジェクトを作成する必要があります。

この白書では、インターシステムズ社の新しい Jalapeño (ハラペーニョ) テクノロジーについて説明します。Jalapeño テクノロジーでは、アプリケーションの開発アプローチに、「Caché-out (Caché から Java)」ではなく、「Java-in (Java から Caché)」の方法を取ります。Caché 内で、POJO (Plain Old Java Object)に永続性を与える方法や、データ・ストレージにオブジェクト技術を使うことの利点について解説します。また、リレーショナルデータベースに対して、Jalapeño テクノロジーを使ってアプリケーションを作るのに必要な手順を概説します。

Jalapeño テクノロジーとは？

インターシステムズ社の Jalapeño テクノロジー(マッピング無しで Java 言語に永続性を与える技術: **J**AvA **L**Anguage **P**ersistense with **N**O mapping) とは、O/R マッピングや、Caché Studio を使わずに、どんな Java の開発環境でもオブジェクトクラスを定義し、定義されたクラスに、Caché 内で永続性を与えることができます。Java クラスとは独立して、Caché によって高パフォーマンスのオブジェクトデータへのアクセスメソッドが自動生成されるので、開発者はアプリケーションで POJO を使うことができます。また、データを格納する方法について考慮する必要がありません。

オブジェクトを永続 Java オブジェクトとして扱い、O/R マッピングが不要になると、開発に要する時間が劇的に減少します。また、データを格納し、取り出す時に、オブジェクトをディスアセンブル / アセンブルする手間が省けるので、パフォーマンスも上がる、という利点があります。

データをオブジェクトとして格納する方が望ましいとはいえ、Java アプリケーション、あるいはサードパーティのデータ分析やレポートツールでは、SQL でのデータベースに問合せが必要な場合があります。Caché の統一データアーキテクチャでは、データを自動的にリレーショナルテーブルとして見せることができます。もちろん、マッピングは不要です。Caché は JDBC をサポートしており、アプリケーション内の SQL クエリは、Jalapeño テクノロジーのオブジェクトの永続メソッドと同じデータベース接続を共有します。

Jalapeño テクノロジーの活用

Caché 内で POJO に永続性を与えるには、2つの方法があります。1つは、POJO 定義から Caché クラスを作成し、コンパイルする方法。2つ目は、Java アプリケーション内に Jalapeño オブジェクトマネージャを組み込む方法です。

Caché クラスの作成

Jalapeño テクノロジーは、Java のクラス定義に基づいて、永続性のある Caché クラスを生成、コンパイルする機能(スキーマ・ビルダと呼ばれる)を持っています。(スキーマ・ビルダは CacheDB.jar 内の Java クラスに含まれています) 開発者は、POJO のクラス定義において”アノテーション”を付加することにより、Caché がオブジェクトモデルを構築する方法をコントロールできます。アノテーション(JDK1.5 で採用されています)は、Java クラスのランタイムの振る舞いには影響を受けません。単純に、要求される Caché オブジェクトモデルについて、メタデータを付加します。例えば、アノテーションは、他のクラスから、ある一つのクラスを継承している、もしくは1つのクラス属性に基づいてインデックスが作成される、といった情報を Caché に伝えます。

例として、1A、1B(次ページ参照)では、簡単な Java のクラス定義と、そこから生成される Caché クラスの例を取り上げています。ここでは、Caché クラスは、インターシステムズ社が提供している %Library.Persistent クラスから継承し、インデックスは “Name” 属性に対して生成していません。

Java 開発者は、スキーマ・ビルダによって生成された Caché クラスを、特に確認したり、それに注意を払う必要は全くありません。また、開発者は、自分の好きな IDE で、POJO を使ってプログラムを書くことができます。オブジェクトモデルが変更された場合は、再度スキーマ・ビルダを動作させるだけで、対応する永続 Caché クラスに変更が反映されます。

例 1A Java クラス

```
import com.intersys.pojo.annotations.CacheClass;
import com.intersys.pojo.annotations.Index;

@CacheClass(name="Person",primaryKey="ID",sqlTableName="PERSON")
@Index(description="Name Index on Person
table",name="PersonIndexOne",propertyNames={"name"},sqlName="PersonIDX")
public class Person {

public String name;
public String ssn;
public String telephone;

}
```

例 1 B 対応する Caché クラス

```
Class User.Person Extends %Library.Persistent [ ClientName = Person, Not ProcedureBlock,
SqlTableName = PERSON ]
{
Property name As %Library.String(JAVATYPE = "java.lang.String", MAXLEN = 4096);

Property ssn As %Library.String(JAVATYPE = "java.lang.String", MAXLEN = 4096);

Property telephone As %Library.String(JAVATYPE = "java.lang.String", MAXLEN = 4096);
```

```
Index PersonIndexOne On name [ SqlName = PersonIDX ];

XData JavaBlock
{
<JavaBlock><Package implementation="CacheRefactor.cache"
pojo="CacheRefactor"></Package><UseSameNames>false</UseSameNames><Name implementation="Person"
pojo="Person"></Name><ResolveNameCollisions>false</ResolveNameCollisions><
EagerFetchRequired>true</EagerFetchRequired></JavaBlock>
}
```

オブジェクトマネージャの使用

Caché に備わっている他の Java バインディング機能で、Caché クラスが Java のプロキシクラスに投影される場合は、永続化メソッド(%Library.Persistent から継承しています)は Java の“アクセサ”メソッドに変換されます。全ての Java クラスは、アクセサ・メソッドのセットを含んでいます。対照的に、Jalapeño テクノロジでは、POJO は、アクセサ・メソッドを含むようには変換されません。その代わりに、Java アプリケーションには、オブジェクトマネージャ、と呼ばれるエレメントを使用します。オブジェクトマネージャは、データベースの接続を確立し、対応する Caché クラスのインスタンスを作成し、アクセサ・メソッドを作成、実行させます。

オブジェクトマネージャは、Java クラスであり、インターシステムズ社の Caché DB.jar の一部として提供されています。使いこなすのに必要な手順は、Java プログラマにとっては既によく行う手順です。

- アプリケーションの、CLASSPATH 宣言に Caché DB.jar を含める
- “POJO”パッケージ (CacheDB.jar には、インポートすべき他のパッケージも入っている可能性もあります)をインポートする
- オブジェクトマネージャのインスタンスを作成

例2では、オブジェクトマネージャのインスタンス作成を紹介しています。オブジェクトマネージャは、データベースサーバへ JDBC での接続を提供します。Caché を使うと、オブジェクトと JDBC からのデータベースアクセスは、同じ接続を共有します。従って、オブジェクトマネージャは、高パフォーマンスの永続メソッドを使うことができますが、データには、SQL を使っても問合せができます。

例2 オブジェクトマネージャのインスタンス作成

```
public DBService (String[] args)
throws Exception
{
    String host = "localhost";
    String username="_System"; // null for default
    String password="sys"; // null for default

    for (int i = 0; i < args.length; i++)
        if (args[i].equals("-user"))
            username = args[++i];
        else if (args[i].equals("-password"))
            password = args[++i];
        else if (args[i].equals("-host"))
            host = args[++i];

    String url="jdbc:Cache://" + host + ":1972/USER";

    Class.forName ("com.intersys.jdbc.CacheDriver");
    Connection connection = DriverManager.getConnection (url, username, password);
    objectManager = ApplicationContext.createObjectManager (connection);
}
```

リレーショナルデータベースの開発

オブジェクトとリレーショナルデータベースアクセスの両方を可能にするオブジェクトマネージャは、Jalapeño テクノロジーで作られた Java アプリケーションが、Caché よりも、リレーショナルデータベースで開発される時に、能力を発揮します。リレーショナルのアーキテクチャでの開発はとても簡単で、たった2つの手順が追加されるだけです。

初めに、適当なリレーショナルデータベースのスキーマを作成します。Caché は、オブジェクトモデルをプロジェクトするエクスポートユーティリティ (CachéDB.jar の一部分) を提供しています。エク

サポートユーティリティは、元は POJO のクラス定義から生成されたものですが、オブジェクトモデルを、リレーショナルデータベースにインポートする DDL ファイルとして、投影することがききます。これは、Caché における、標準のリレーショナルプロジェクションではないことを注記します。Caché のオブジェクトスキーマは、Java クラス定義から生成され、エクスポート機能は POJO と関連付けられるので、リレーショナルデータのスキーマを作る際には、その情報を利用することができるからです。

一旦、適切なリレーショナルデータベースのスキーマが作られると、残る作業は、Caché の代わりにリレーショナルデータベースに接続するための、オブジェクトマネージャの設定です。オブジェクトマネージャは、自動的に Caché に接続する時はオブジェクトの永続化メソッド (Open, Save, New, Delete) を使い、リレーショナルデータベースに接続する時はリレーショナル永続化メソッド (Select, Update, Delete) を使います。

インターシステムズ社の Jalapeño テクノロジは、リレーショナル環境における Java アプリケーション開発を簡単にしますが、Caché で開発すれば、更にアプリケーションを速く動作させることができます。Caché は永続オブジェクトに対して高いパフォーマンスを実現するだけでなく、特に複雑な SQL クエリに対しても、リレーショナルデータベースより高速に応答するからです。

結論

Caché は、JDBC とオブジェクトデータアクセスによって、Java アプリケーションに対して、データの永続性を与える点については、長きに渡っていくつかの方法を提供してきました。しかし、これまでは、そのアプローチはあくまでも「Caché 中心」でした。開発者は Caché でのオブジェクト定義を行った後に、Java の環境へ投影させることが必要でした。

Caché の新しい Jalapeño テクノロジは、開発者にデータの永続性を実現するための方法として、“Java-in” という選択肢を提供します。POJO のクラス定義から、永続 Caché クラスが定義され、コンパイルすることができます。ランタイムにおいては、データベースの接続性と永続性は、Jalapeño の一部分として備わっているオブジェクトマネージャが処理します。開発者は、データベース上、どのように情報に永続性が与えられるか考えることなく、オリジナルの POJO を使うことができます。

Caché IDE を使う必要がなくなったのに加え、O/R マッピングに時間を費やさずに済みます。Caché は、同じ接続で、オブジェクトアクセスとリレーショナルアクセスの両方を実現するので、開発者はオブジェクトのみに注力していれば良いのです。Java アプリケーションは高パフォーマンス

でオブジェクト指向の永続化メソッドを使い、必要に応じて Caché データベースに対して SQL クエリを実行することもできます。

Jalapeño テクノロジは、開発者が Caché のみでアプリケーション構築をしなければならないといった、制限をするものではありません。結果的にパフォーマンスは期待ほどは良くありませんが、最小限の付加作業で、Jalapeño テクノロジで作られた Java アプリケーションは、リレーショナルデータベース上でも動作します。

* 本白書は、米国インターシステムズ社の”Plain Old Java Persistence With Caché”の日本語訳です。不明な点は、英文本文にてご確認ください。

INTERSYSTEMS

インターシステムズジャパン株式会社

〒160-0023

東京都新宿区西新宿6-10-1 日土地新宿ビル17階

www.intersystems.co.jp

InterSystems Caché 、 InterSystems Ensemble は、米国インターシステムズ社の商標です。

© copyright 2006 InterSystems Corporation. All rights reserved 5-06