

Professional Computing

95 Kč

LISTOPAD 2006 | ROČNÍK VII

RECENZE | NÁZOR | TECHNOLOGIE

TÉMA ČÍSLA

IT SECURITY

Sláva, uznání, sebeuspokojení... Tak nějak vypadal žebříček motivů kybernetických útočníků před mnoha lety. Nehnaly je žádné „vyšší“ pohnutky. Prostě se jen o „něco“ pokoušeli – a když neuspěli, šli klidně zkoušet štěstí o dům dál. Jak ale šel čas, informační technologie postupně prostoupily prakticky všechny oblasti našich životů. Mnozí pochopili, že mohou získat více než jen uznání úzké komunity či pomíjivé sebeuspokojení. Ale že mohou své znalosti proměnit na reálné peníze...

Příloha Professional Computingu

ICT ve zdravotnictví

Přesnost a včasnost informace pro lékaře, stejně jako podpora náročné logistiky provozů nemocnic, ale i laboratoří či ambulancí jsou důležité prvky podporující rozvoj nasazení výpočetní techniky pro potřeby léčebného procesu. Na druhou stranu faktem ale je, že kromě výhod elektronicky uložené informace zatím tyto technologie nepřinesly další principiální výhody. Skeptičtí uživatelé to nazývají problémem „chytrého psacího stroje“ – lékař opustil mechanický psací stroj výměnou za počítač, nicméně styl jeho práce zůstal stejný.

TECHNOLOGIE

Javovské perzistence s Caché

Díky své vlastnosti „jednou to napiš a implementuj kdekoliv“ má programovací jazyk Java mnoho příznivců a uživatelů. Ale Java je také objektově-orientovaný jazyk a jeho datové objekty nebyly autory navrženy jako perzistentní. I nejhrolivější přívrženci Javy připustí, že s objektově-relačním mapováním, vyžadujícím uchování javovských objektů v relační databázi, je spousta práce. Podle některých odhadů může zabrat až 60 až 70 procent úsilí vývojářů...

TÉMA ČÍSLA

IT security



Builder, aby se tyto změny odrazily v odpovídajících perzistentních Caché třídách.

Použití Object Managera

V jiných propojovacích mechanismech Javy a Caché, třeba při vytváření projekcí Caché tříd jako proxy Java tříd, jsou perzistentní metody (zděděné z %Library.Perzistent) transformovány do Java „přístupových“ metod. Každá Java třída obsahuje svou vlastní množinu přístupových metod. Naopak při použití technologie Jalapeño se objekty POJO nemění, aby zahrnuly přístupové metody. Místo toho javovská aplikace používá element zvaný Object Manager, který navazuje spojení s databází, vytváří instance příslušných Caché tříd i přístupové metody, které také spouští.

Object Manager je Java třídou, kterou InterSystems dodává jsou součástí knihovny tříd CacheDB.jar. K jejímu použití je potřeba udělat následující kroky, které jsou důvěrně známé každému programátorovi v Javě:

- Zahrňte CacheDB.jar do proměnné CLASSPATH v aplikaci.
- Importujte balíček „pojo“. (V CacheDB.jar mohou být také další balíčky, které můžete chtít naimportovat.)
- Vytvořte instanci třídy Object Manager.

Příklad 2 ukazuje vytvoření instance Object Manager. Všimněte si, že Object Manager (dále česky „Správce Objektů“) navazuje JDBC spojení s databázovým serverem. V Caché mohou objekty i JDBC databázový přístup sdílet stejné spojení. Tím pádem může Správce Objektů využívat vysoce výkonné perzistentní metody, ale dotazy na data mohou být také prováděny s pomocí SQL.

Příklad 1A – Java třída

```
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", propertyName=[,name], sqlName="PersonIDX")
public class Person {
    public String name;
    public String ssn;
    public String telephone;
}
```

Příklad 1B – odpovídající vygenerovaná třída Caché

```
Class User.Person Extends %Library.Perzistent [
    ClientName = Person, Not ProcedureBlock,
    SqlTableName = PERSON ]
{
    Property name As %Library.String(JAVATYPE =
    „java.lang.String“, MAXLEN = 4096);
    Property ssn As %Library.String(JAVATYPE = „ja-
    va.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=“Cache-
        Refactor.cache“
        pojo=“CacheRefactor“></Package><UseSameNa-
        mes>false</UseSameNames><Name
        implementation=“Person“
        pojo=“Person“></Name><ResolveNameCollisions
        >false</ResolveNameCollisions><
        EagerFetchRequired>true</
        EagerFetchRequired></JavaBlock>
    }
}
```

Příklad 2 – instancie třídy Object Manager

```
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.createObject-
    Manager (connection);
}
```

Implementace nad relační databází

Schopnost Správce Objektů ovládat přístup jak k objektovým, tak relačním datům nabývá zvláštní důležitost, jestliže Java aplikace vytvořená s technologií Jalapeño vyžaduje implementaci nad relační databází, místo Caché. Nasazení v relační architektuře je jednoduché a vyžaduje jen dva do- datečné kroky.

Za prvé musí být vytvořeno příslušné schéma relační databáze. Caché pro tento případ nabízí exportní utilitu (jako součást CacheDB.jar), jež může vytvořit projekci objektového modelu – původně odvozeného z definic tříd POJO – jako DDL souborů, které mohou být importovány do relační databáze. Je důležité si povšimnout, že nejde o stejnou věc jako v případě standardních relačních projekcí

Caché. Protože objektové schéma v Caché bylo vytvořeno z definic Java tříd, exportní utilita „ví“ pár věcí o objektech POJO a používá tyto informace při vytváření schématu relačních dat.

Jakmile bylo příslušné relační databázové schéma vytvořeno, stačí jen nakonfigurovat Správce Objektů tak, aby se připojoval k relační databázi namísto Caché. Správce Objektů automaticky použije metody objektové perzistence (Open, Save, New, Delete) při připojování ke Caché a metody relační perzistence (Select, Update, Insert, Delete) při připojování k relační databázi.

Ačkoliv technologie Jalapeño společnosti InterSystems usnadňuje nasazení javovských aplikací v relačním prostředí, vývojáři pravděpodobně zjistí, že aplikace běží rychleji s Caché. Caché totiž nejen že umožňuje vysoce výkonnou objektovou perzistenci, ale bylo také dokázáno, že odpovídá na SQL dotazy – a to zvláště ty složitější – rychleji než relační databáze.

SHRNUTÍ

Caché dlouhou dobu podporovala několik způsobů datové perzistence v Java aplikacích, přes JDBC, nebo přístup k objektovým datům. Ale až doposud byly tyto postupy „Caché-centrické“. Po vývojářích se chtělo, aby definovali objekty v Caché a pak vytvořili jejich projekci v prostředí Javy. Nová technologie Jalapeño v Caché dává vývojářům možnost pro dosažení datové perzistence použít postup „Java-in“. Perzistentní Caché třídy mohou být definovány a kompilovány z definic tříd POJO. Při běhu programu jsou databázové připojení a perzistence řízeny Správce Objektů, který je poskytován jakou součástí Jalapeño. Vývojáři mohou použít originální objekty POJO, aniž by přemýšleli o tom, jak je v databázi zajištěna perzistence. Kromě toho, že jsou vývojáři v Javě osvobozeni od nutnosti používat Caché Studio, jsou také osvobozeni od únavného a časově náročného objektově-relačního mapování. Caché dovoluje jak objektový, tak relační přístup k datům jedním spojením, takže vývojáři mohou uvažovat v pojmech objektů. Jejich javovské aplikace mohou využívat vysoce výkonné metody objektově-orientované perzistence a dotazovat Caché databázi s použitím SQL, pokud se to hodí. Technologie Jalapeño neomezuje vývojáře v implementaci jejich aplikací nad Caché. Při minimální další práci může javovská aplikace vytvořená s technologií Jalapeño běžet nad relační databází, i když výsledný výkon pravděpodobně nebude tak skvělý.

Andreas Dieckow je Principal Product Manager a Strategic Planning společnosti InterSystems Corporation. Zastihnout ho můžete na adrese andreas.dieckow@intersystems.com