CODE GEAR FROM Borland Developer Camp

【B6】チュートリアルセッション

「LiveSourceを使ったモデリング開発」

CodeGear
高橋智宏

CODE GEAR FROM Borland Developer Camp

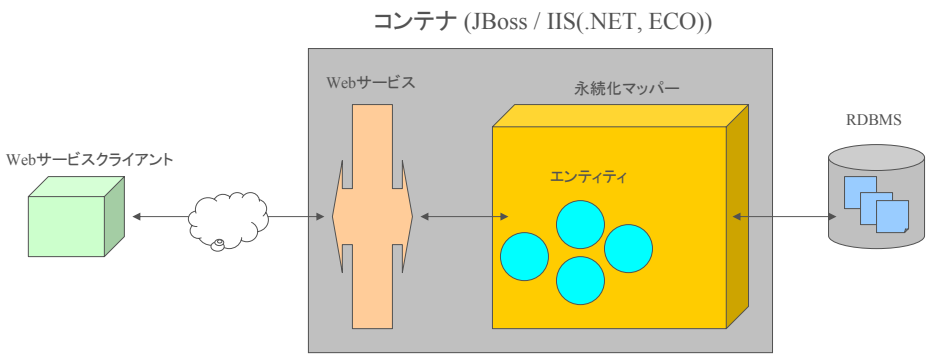
アジェンダ

- 今回作成するシステムの概念図
- JGear LiveSource for Eclipse 3.2
- JBoss 4 を使用した場合の開発手順
- RAD Studio 2007（Delphi .NET）
- .NET + ECO を使用した場合の開発手順

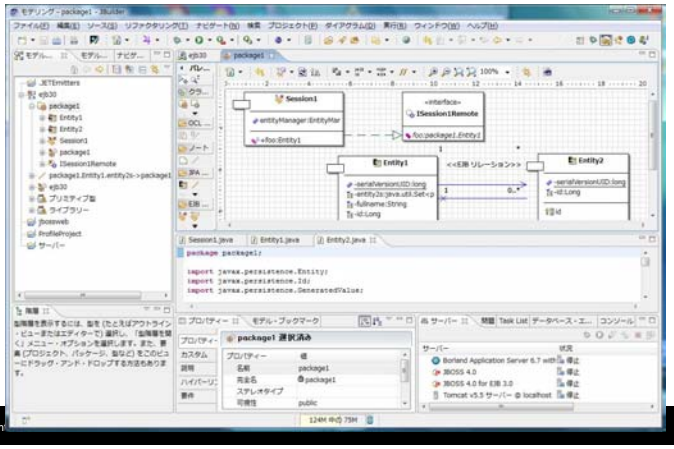
今回作成するシステムの概念図

RDBMS
エンティティ
永続化マッパー
コンテナ
Webサービス

Webサービスでエンティティを操作



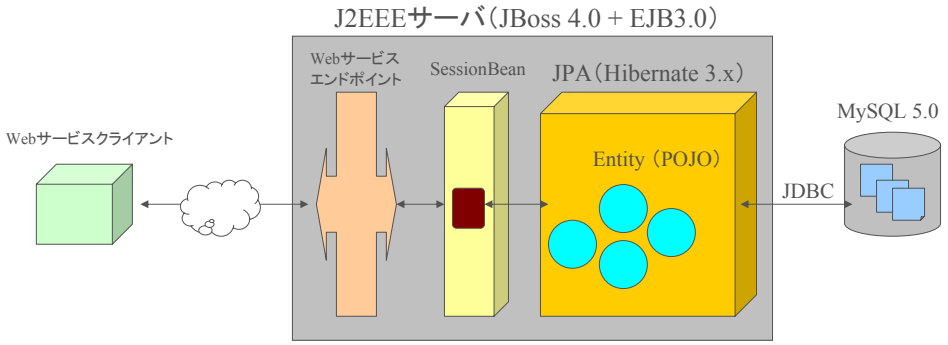
JGear LiveSource for Eclipse 3.2



JGear LiveSource

- Eclipse 3.2 向けのプラグイン
- JBuilder 2007 のJ2EE開発機能を単体でも利用可能にしたもの
- モデルとソースコードやアノテーションが自動同期
- JBoss 4 などのJ2EEサーバが同梱され、簡単に統合が可能
- カタログ
  - http://www.codegear.com/article/36933/images/36933/jgear_livesource_datasheet_ja.pdf

JBoss 4 を使用した場合の開発手順



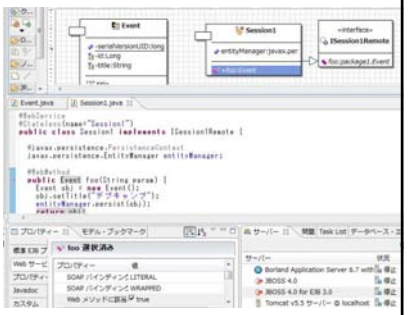
JBoss 4 を使用した場合の開発手順（続き）

- MySQLのセットアップ
  - 今回は utf-8 のテーブルを使用
- JBoss 4.0 の設定
  - JDBCドライバの追加
  - データソース(.xml)の追加
- IDEとJBossを統合
  - EJB 3.0 を選択
  - インストール済みサーバー・ランタイムに JBoss を追加
  - サーバーペインに JBoss を追加
- データベース・エクスプローラにMySQLを追加

CODE GEAR FROM Borland Developer Camp

JBoss 4 を使用した場合の開発手順（続き）

- EJBモデリングプロジェクトを作成
- JPAのEntity（例:Event）を作成
  - プロパティの追加
- Stateless Session Beanを作成
  - EntityManagerへの参照を追加
  - ビジネスメソッドの追加および実装
- SessionBeanをWebサービスとして公開
  - 実装クラスおよびメソッドにアノテーションを設定



CODE GEAR FROM Borland Developer Camp

JBoss 4 を使用した場合の開発手順（続き）

- 永続化ディスクリプタの設定
  - persistence.xml
- プロジェクトのビルド
- JBossへのデプロイおよびJBossの起動
- MySQLへのeventテーブルが自動で作成される

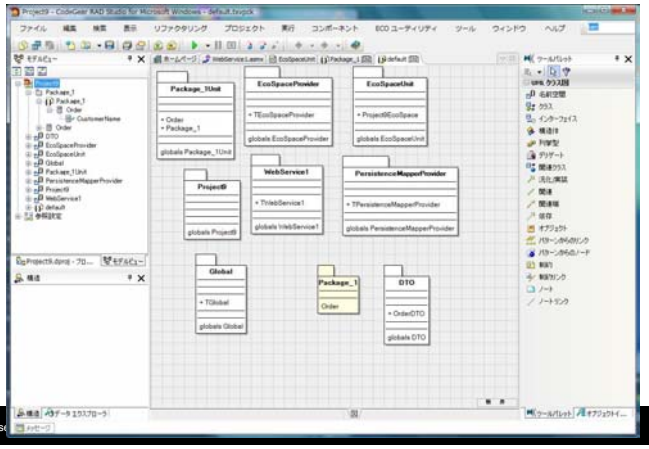
```
| event | CREATE TABLE `event` (
  `id` bigint(20) NOT NULL auto_increment,
  `title` varchar(255) default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
```

- WebサービスのWSDLファイルが自動で生成される
  - file:/C:/JBuilder2007/thirdparty/jboss-4.0.5.GA/server/default/data/wsdl/test.jar/Session1Service54632.wsdl

テストおよび拡張

- WSDLファイルを使用して、Webサービスクライアントを作成
  - MySQLのテーブルにデータが格納されることを確認
- エンティティ同士のリレーションシップ
  - エンティティリレーションシップの追加とソースコードの同期
  - テーブルや外部キーの自動生成
  - 例: One to Many

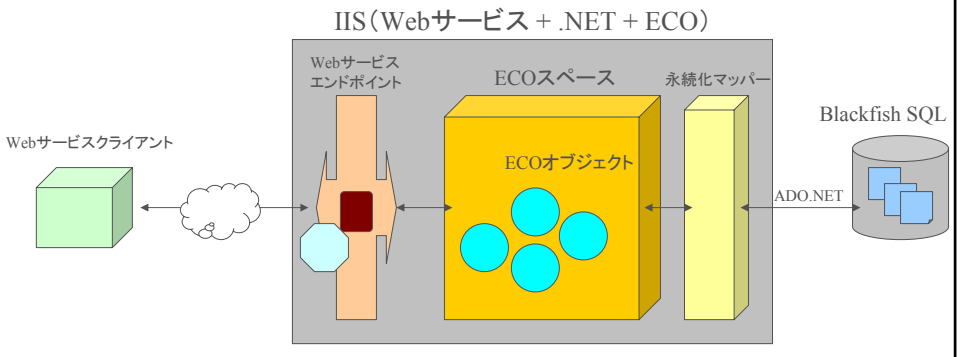
RAD Studio 2007（Delphi .NET）



RAD Studio 2007（Delphi .NET）

- .NET 2.0 をサポートしたDelphi言語向け統合開発環境
  - ASP.NET 2.0
  - Webサービス
  - ADO.NET 2.0
- Architect版にのみ、ECO（Enterprise Core Objects）が付属
- Blackfish SQL for .NET が標準添付され、配布も可能
- モデルとソースコードが自動同期
- 簡易Webサーバ Cassini が同梱され、すぐに開発が可能
- カタログ
  - http://www.codegear.com/article/36908/images/36908/radstudio2007_datasheet_ja.pdf

.NET + ECO を使用した場合の開発手順



CODE GEAR FROM Borland Developer Camp

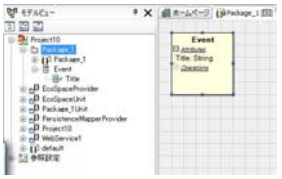
.NET + ECO を使用した場合の開発手順（続き）

- Blackfish SQL for .NET のセットアップ
  - データエクスプローラの「BlackfishSQL Remote Provider」に追加
  - データベースファイル(.jds)の作成
- ECOプロジェクトの作成
  - 「Eco Delphi .Net プロジェクトウィザード」を起動
  - 「ASP.NET」の「Webサービス」を選択
  - 永続化ストレージに「Blackfish SQL」を選択
- EcoSpaceProvider.pas をテンプレートからコピーして追加
  - ECOの不具合のため。今後のアップデートで修正される予定。
  - クラス名などを一括変換
- .asmx ファイルの Class属性を正しいクラス名に修正
  - ECOの不具合のため。今後のアップデートで修正される予定。

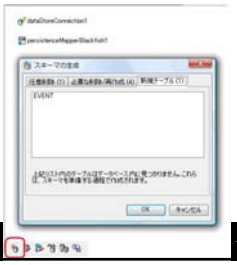
CODE GEAR FROM Borland Developer Camp

.NET + ECO を使用した場合の開発手順（続き）

- モデルとしてECOクラスをクラス図に追加
  - 例：Eventクラス
  - 「属性」の追加



- 永続化マッパー内の接続コンポーネントを設定
  - PersistenceMapperProvider.pas
  - DataStoreConnectionコンポーネントのConnectionStringプロパティ
    - アウトプロセスとしてのBlackfish SQLに接続
- 永続化マッパーでテーブルの自動生成
  - 事前にプロジェクトのビルドが必要



.NET + ECO を使用した場合の開発手順（続き）

- 新規namespace内にデータ転送用クラスを再定義
  - EJB 3.0 の Entity とは違い、ECOオブジェクトはECOスペースと結合した状態のみで利用可能
  - ECOオブジェクトは、Webサービスのパラメータとしては使用できない
- Webサービスメソッドの追加

```
function TWebService1.NewEvent(Name: String): EventDTO;
var
  anEvent: Event;
begin
  anEvent := Event.Create(EcoSpace);
  anEvent.Title := Name;
  UpdateDatabase;

  Result := EventDTO.Create;
  Result.Title := anEvent.Title;

  DoneWithEcoSpace;
end;
```


テストおよび拡張

- Webサーバーを起動
  - 「開始ページ」の指定
  - 「Cassini」が自動起動
- WebブラウザでWebサービスを確認

- エンティティ同士のリレーションシップ
  - エンティティリレーションシップの追加とソースコードの同期
  - テーブルの自動生成
  - 例: One to Many

Q&A

Any question?