【B3】Delphiテクニカルセッション

EMBARCADERO TECHNOLOGIES

DEVELOPER CAMP

単層から多層アーキテクチャへの遷移
dbExpress DataSnap 2009を利用するデータベースアプリケーションの実装とは

フリーランス
山本聡

アジェンダ、プレゼンテーションの概要

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

- 2層C/Sアプリケーションの紹介。そして問題点。
- 3層C/Sアプリケーションによって解決される事
- BDEからdbExpressへの移行
- dbExpressとDatasnapの連携での3層アプリケーション
- D2009での開発メリット。新機能紹介。とても便利なサーバーメソッド。

今日の目標

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

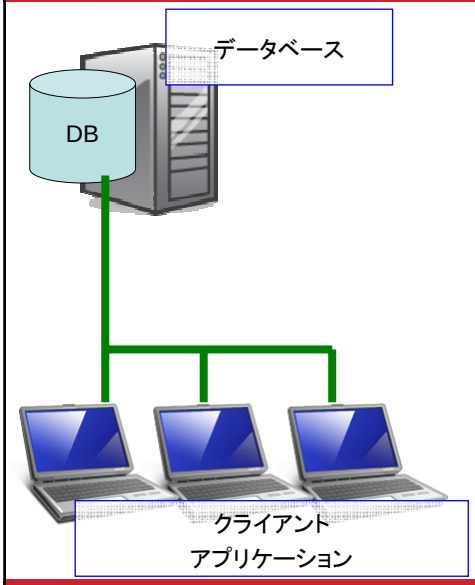
- BDEをいつでも乗り換えられるようになっていて欲しい。移行は簡単です。
- 3層C/Sアプリケーションのメリットを知り、その開発がDelphiならとても楽に可能だということを知って欲しい。
- D2009のサーバーメソッドの威力を存分に使って、負荷を適所に配分する、負荷分散処理までも視野にいれたネットワークシステムを構築していって欲しい。

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

2層C/Sアプリケーションの現状　そして問題点

2層C/Sアプリケーションの現状　そして問題点

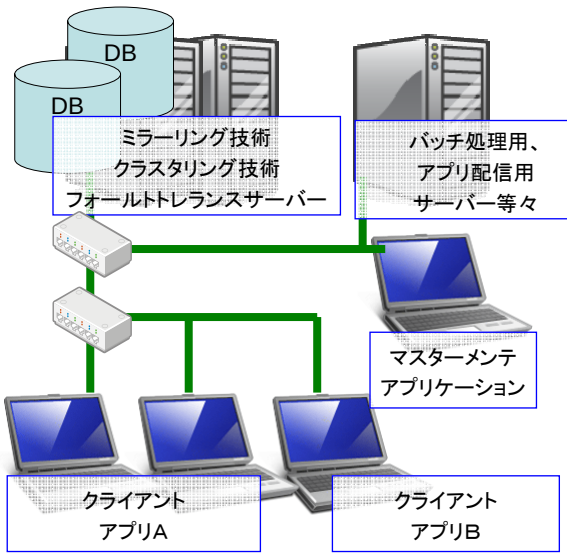
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- かつては単独PC上で動いていたものが、DBと分離することによって、情報共有できるシステムとして、非常に重要な価値を生み出しています。
- 基本的には、世間の多くのC/Sシステムはこの形式で動いています。

2層C/Sアプリケーションの現状　そして問題点

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

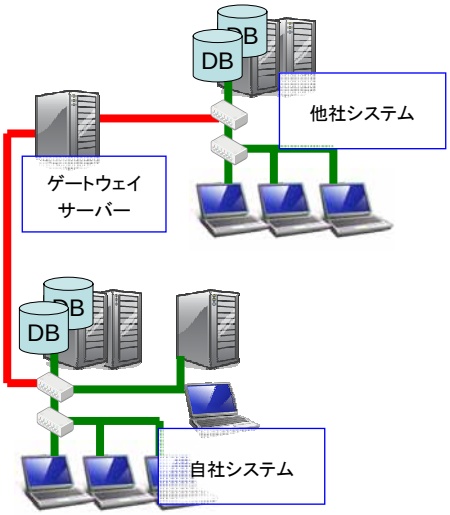


- 2層C/Sシステムは現在、非常に複雑化してきました。
- システムは使い勝手のよいように自由自在に拡張されてきました。
- もはや通信をDB接続だけでは行っていないはずです。

  クライアント間でのメッセージ即時通知の為のソケット通信や、レポート作成したOfficeファイルの共有フォルダ等々が行われています。

2層C/Sアプリケーションの現状　そして問題点

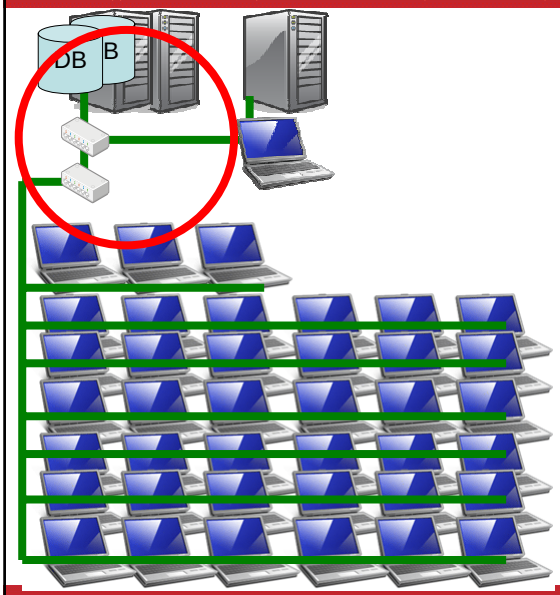
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- このように複雑なシステムが、さらに他社システムと連携する場合もあります。
- 自社システムの信頼性を確保する為に、他社システムとネットワークを切り分ける場合もあるでしょう。
- その他社とも情報は連携しなければならないために、ゲートウェイサーバーをおいて、情報をやりとりするプログラムが作られています。
- サーバールームは大規模化してきています。
- システムはより複雑により自由度が高い方向に進化し続けています。
- そして2層というところだと、負荷が耐えられない部分が存在してきます。

2層C/Sアプリケーションの現状　そして問題点

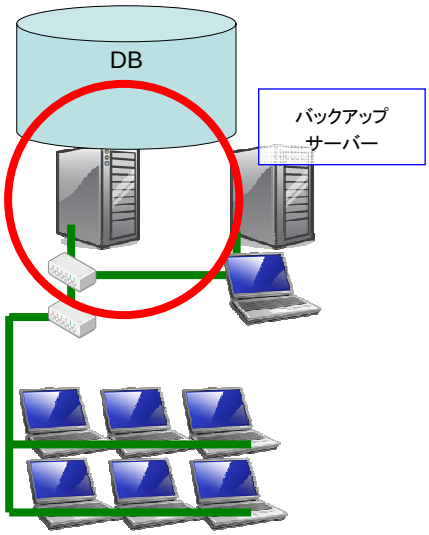
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- 業務が拡大して、クライアントがどんどん増加して、当初予定していた以上にシステムに高負荷がかかる場合があります。
- 通信量が問題なのかDBサーバーへの負荷が過大すぎるのか、
- すぐに対策はうちにくいかもしれません。
- **あと、クライアント配布がめんどくさいです。**

2層C/Sアプリケーションの現状　そして問題点

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



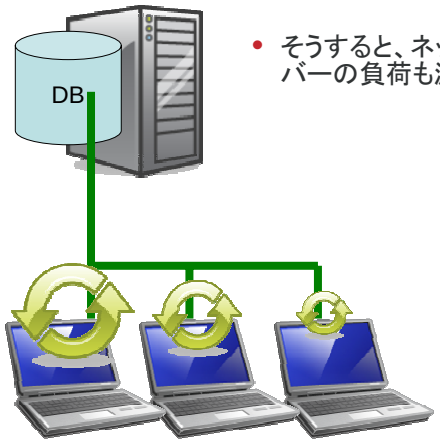
- 予想以上のデータの増加によるDB肥大化も起こりえます。
  ※事前に予測していない場合もあったり。
- DBバックアップ時の巨大バックアップファイルの転送負荷や、DBアクセスにともなってネットワークトラフィックやサーバー負荷の問題がでてきます。

2層C/Sアプリケーションの現状　そして問題点

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

解決策は？

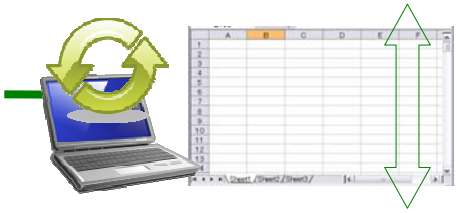
- DBアプリケーションの開発で心がけることは通信量を減らす。そのDB接続待機時間を減らすこと
- そうすると、ネットワークトラフィックも、DBサーバーの負荷も減ることが予測される。



2層C/Sアプリケーションの現状　そして問題点

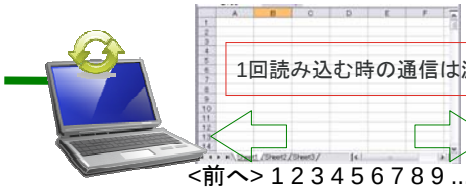
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

解決策は？、通信量を減らす…



- 全てのデータを持ってくる形の使いやすいユーザーインターフェースで
- 全データを閲覧可能で、フィルタリングやソートも自在だったものが・・・

- 使いにくいクリック切り替え式になるかもしれません。切替時の画面遷移は遅い可能性もあります。



1回読み込む時の通信は減った

<前へ> 1 2 3 4 5 6 7 8 9 ... <次へ>

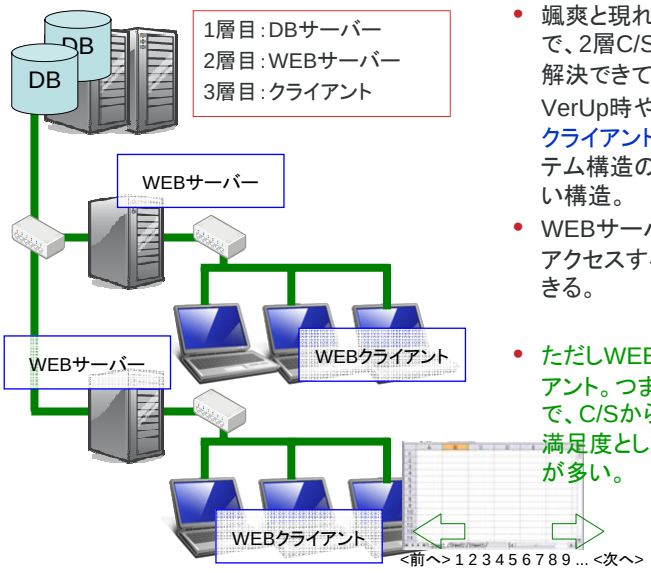
- けれども・・・
- 改造は簡単ではなく、大幅な改造になるかもしれない・・・
- どうせ大きく改造するならば、ネットワーク構成からシステムを見直せないだろうか。

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

3層C/Sアプリケーションによって解決される事

3層アプリケーション、WEBシステムの登場

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

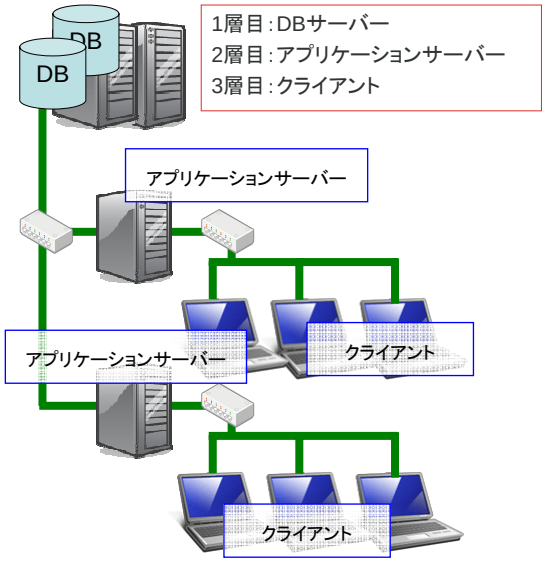


- 颯爽と現れたWEBアプリケーションで、2層C/Sシステムの一部の問題は解決できてしまった。
  VerUp時や、DB障害、切り替え時、クライアント配布が必要ない。※システム構造の大幅な変更にも耐えやすい構造。
- WEBサーバーだけがDBサーバーにアクセスすることで負荷分散も考慮できる。

- ただしWEBはそもそも非リッチクライアント。つまり、プアクライアントなので、C/SからWEBに移行すると、顧客満足度としては、低くなってしまう場合が多い。

3層C/Sアプリケーションによって解決される事

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

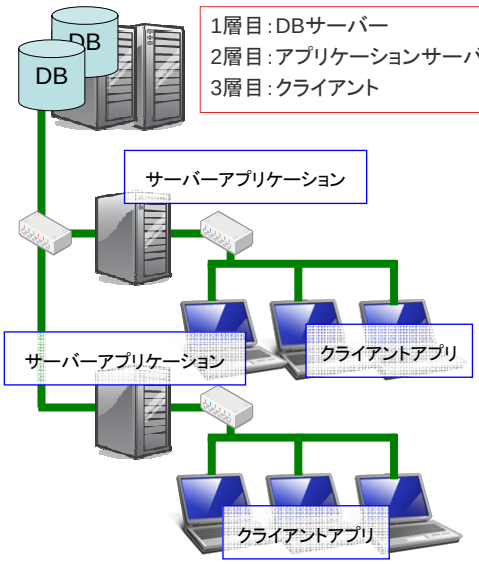


- ある程度、改造、改良、新設計を行うなら、DBの接続時のデータ読み込みの量を調整するだけではなく、ネットワークシステムとしての、WEBアプリケーションの3層構造のいいとこどりをしたい。。

- 今までは簡単に実装するのは難しそうだったが。今度のDelphi2009ならほとんどコードを書く必要がなくコンポーネントで対応してくれる。

3層C/Sアプリケーションによって解決される事

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- WEBアプリの用にクライアントの配布問題が解消はされないと思います。
  ※スマートクライアントの仕組みを作るのは比較的面倒なことではないと思いますが、自作する必要はありそうですね。
- **解決される事は。**
- 自由度が非常に高くなり、負荷が分散される要素がたくさんです。クライアント側で処理すべきことや、アプリケーションサーバー側で処理すべき事がわかれば、それを振り分ける事ができるでしょう。
- 例えば、帳票のPDFの事前生成などの負荷分散に便利
- サーバーサイドのメンテナンスが楽になりそうです。

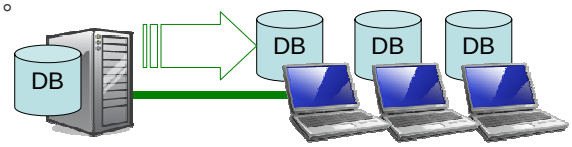
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

BDEからdbExpressへの移行

BDEからdbExpressへの移行

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

- なぜ行うのか。
  - BDEの構造は常にクライアントキャッシュ型の動作。起動時にDBの内容を全てBDE側までとってくる設計のために、非常に重くなってしまう。
  - dbExpressは、クライアントキャッシュ型でもライトなデータ受信型でもどちらも対応可能。



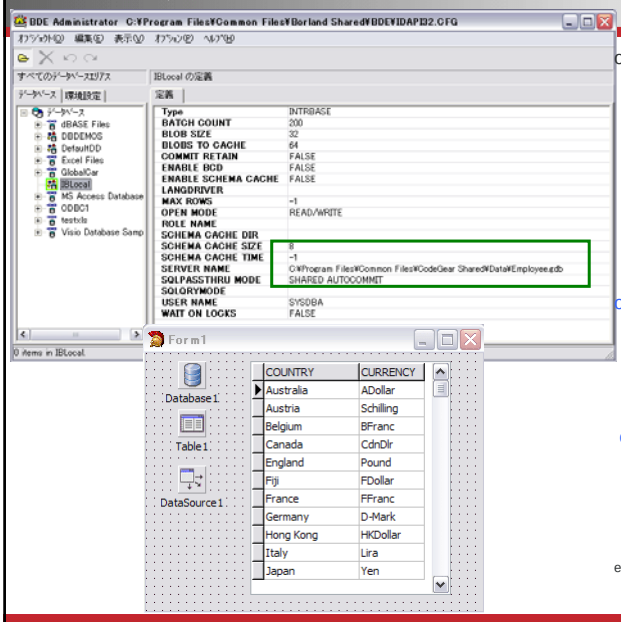
  - BDE自体の配布を行う必要があり、BDEを操作してエイリアスを設定するため手間が発生する。※アプリケーション側でも設定できるようですが、特殊なようです。
  - dbExpressはDB接続ドライバなどが、すべてexeに吸収されるために、全く配布上の問題点を抱えない。

- ※今更ながら BDE (Borland Database Engine)

http://homepage1.nifty.com/ht_deko/tech024.html#tech053

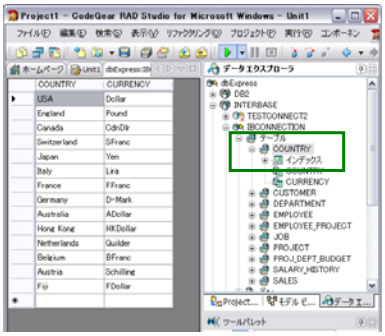
BDEアプリケーション

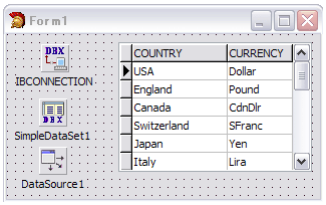
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP




```
object Form1: TForm1
  object Database1: TDatabase
    AliasName = 'IBLocal'
    Connected = True
    DatabaseName = 'dbname'
    LoginPrompt = False
    Params.Strings = (
      'USER NAME=SYSDBA'
      'PASSWORD=masterkey')
    SessionName = 'Default'
  end
  object Table1: TTable
    Active = True
    DatabaseName = 'dbname'
    SessionName = 'Default'
    TableName = 'COUNTRY'
  end
  object DataSource1: TDataSource
    DataSet = Table1
  end
  object DBGrid1: TDBGrid
    DataSource = DataSource1
  end
end
```

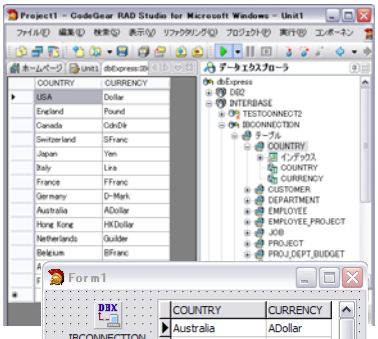
dbExpressアプリケーション

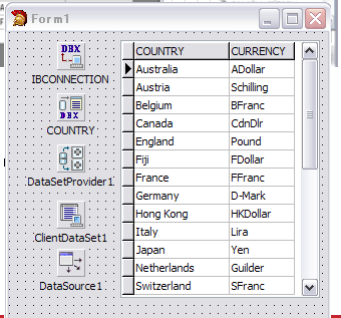





```
object Form1: TForm1
 object IBCONNECTION: TSQLConnection
  ConnectionName = 'IBCONNECTION'
  DriverName = 'Interbase'
  GetDriverFunc = 'getSQLDriverINTERBASE'
  LibraryName = 'dbxint.dll'
  LoginPrompt = False
  Params.Strings = (
   'DriverName=Interbase'
   'Database=C:¥Borland¥InterBase¥examples¥database¥employee.gdb'
   'RoleName=RoleName'
   'User_Name=sysdba'
   'Password=masterkey'
   'ServerCharSet='
   'SQLDialect=3'
   'ErrorResourceFile='
   'LocaleCode=0000'
   'BlobSize=-1'
   'CommitRetain=False'
   'WaitOnLocks=True'
   'Interbase TransIsolation=ReadCommited'
   'Trim Char=False')
  VendorLib = 'GDS32.DLL'
  Connected = True
 end
 object SimpleDataSet1: TSimpleDataSet
  Active = True
  Aggregates = <>
  Connection = IBCONNECTION
  DataSet.Active = True
  DataSet.CommandText = 'COUNTRY'
  DataSet.CommandType = ctTable
  DataSet.DataSource = DataSource1
  DataSet.MaxBlobSize = -1
  DataSet.Params = <>
  Params = <>
 end
 object DataSource1: TDataSource
  DataSet = SimpleDataSet1
 end
 object DBGrid1: TDBGrid
  DataSource = DataSource1
 end
end
```


dbExpressアプリケーション






```
object Form1: TForm1
 object IBCONNECTION: TSQLConnection
  ConnectionName = 'IBCONNECTION'
  DriverName = 'Interbase'
  GetDriverFunc = 'getSQLDriverINTERBASE'
  LibraryName = 'dbxint.dll'
  LoginPrompt = False
  Params.Strings = (
   'DriverName=Interbase'
   'Database=C:¥Borland¥InterBase¥examples¥database¥employee.gdb'
   'RoleName=RoleName'
   'User_Name=sysdba'
   'Password=masterkey'
   'ServerCharSet='
   'SQLDialect=3'
   'ErrorResourceFile='
   'LocaleCode=0000'
   'BlobSize=-1'
   'CommitRetain=False'
   'WaitOnLocks=True'
   'Interbase TransIsolation=ReadCommited'
   'Trim Char=False')
  VendorLib = 'GDS32.DLL'
  Connected = True
 end
 object COUNTRY: TSQLDataSet
  CommandText = 'COUNTRY'
  CommandType = ctTable
  DbxCommandType = 'Dbx.Table'
  MaxBlobSize = -1
  Params = <>
  SQLConnection = IBCONNECTION
 end
 object DataSetProvider1: TDataSetProvider
  DataSet = COUNTRY
 end
 object ClientDataSet1: TClientDataSet
  Active = True
  Aggregates = <>
  Params = <>
  ProviderName = 'DataSetProvider1'
 end
 object DataSource1: TDataSource
  DataSet = ClientDataSet1
 end
 object DBGrid1: TDBGrid
  DataSource = DataSource1
  TabOrder = 0
 end
end
```

対応コンポーネント

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

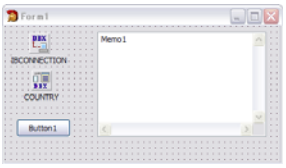
- 非常におおざっぱですが

	BDE	dbExpress
接続系	TDatabase 接続補助	TSQLConnection
DataSet系	TTable/TQuery等	TSimpleDataSet
	TDataSource	同じ
	TDBGridなど	同じ

- TSimpeDataSet
  - 実は複数コンポーネントの機能を合算したもの。
  - dbExpressはスピード重視のため、DataSetは単方向つまりRecordSetでいうnext方向にしかデータベースサーチを行わないようになっている。
  - そのDataSetはTSQLDataSet/TSQLQuery/TSQLTable/TSQLStoredProcが存在するが、単方向のためDBGridなどとは直接の接続ができない。
  - そこで、DataProviderとClientDataSetを用いてDBGridと接続する。ClientDataSetは内部にキャッシュを持つ構造。
  - SQLConnection-SQLDataSet-DataProvider-ClientDataSet-DataSource
    これを簡略化して次のようにしたもの。
    SQLConnection-SimpleDataSet-DataSource
  - これを使えば、BDEと同じ数のコンポーネントを使用することになる。

dbExpressアプリケーション

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP




```
procedure TForm1.Button1Click(Sender: TObject);
var
  I: Integer;  S: String;
begin
  Memo1.Lines.Clear;

  COUNTRY.Close;
  COUNTRY.Open;
  S := '';
  for I := 0 to COUNTRY.FieldCount - 1 do
    S := S + ' ' +  COUNTRY.Fields[I].FieldName;
  Memo1.Lines.Add(S);
  while not COUNTRY.Eof do
  begin
    S := '';
    for I := 0 to COUNTRY.FieldCount - 1 do
      S := S + ' ' + COUNTRY.Fields[I].AsString;
    Memo1.Lines.Add(S);
    COUNTRY.Next;
  end;
end;
```

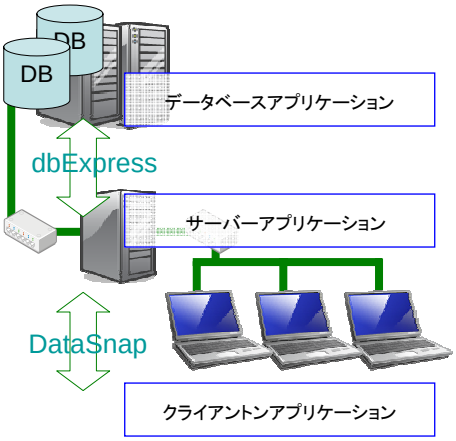
- 単にSQLを送信してレコードを読み込むなら、とてもシンプルに組む事ができます。使用コンポーネントも2つだけ。
- 高速に動作します。

EMBARCADERO TECHNOLOGIES

DEVELOPER CAMP

dbExpressとDatasnapの連携での3層アプリケーション

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

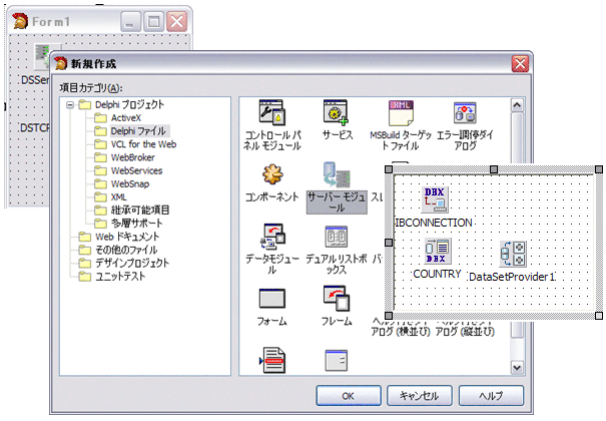


1層目：DBサーバー
2層目：アプリケーションサーバー
3層目：クライアント

- 多層といいつつ、ここでは3層ですが。
- DBとサーバーアプリケーション間はdbExpress
- サーバーアプリとクライアントアプリとの間はDataSnap

DataSnapサーバーアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



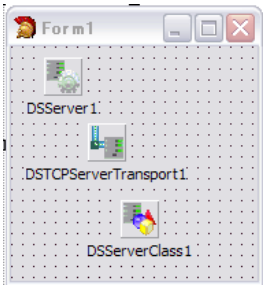
プロジェクトを新規作成して、新規作成で、サーバーモジュールを作成します。

そのサーバーモジュールがTDSServerModule。すなわちDataSnapです。

DSServerModule1上にTSQLConnectionとTSQLTableとTDataSetProviderを配置しておきます。(データエクスプローラからドラッグ &ドロップ)

DataSnapサーバーアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



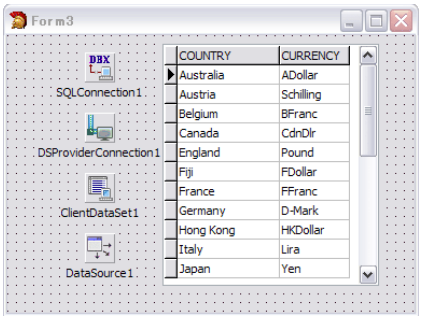

```
object Form1: TForm1
  object DSServer1: TDSServer
    AutoStart = True
    HideDSAdmin = False
  end
  object DSTCPServerTransport1:
      TDSTCPServerTransport
    PoolSize = 0
    Server = DSServer1
    BufferKBSize = 32
    Port=211
  end
  object DSServerClass1: TDSServerClass
    Server = DSServer1
    OnGetClass = DSServerClass1GetClass
    LifeCycle = 'Session'
  end
end
```

- サーバーアプリケーションのメインフォーム側にもDataSnapコンポーネントを配置します。
- イベントハンドラも記述してください。
- イベントハンドラを記述します。

```
//uses で DSServerModule側ユニットを追加しておく。
procedure TForm1.DSServerClass1GetClass(DSServerClass:
    TDSServerClass;
  var PersistentClass: TPersistentClass);
begin
  PersistentClass := TDSServerModule1;
end;
```

DataSnapクライアントアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- クライアントアプリは、プロパティを設定するだけで、サーバーアプリと接続されます。
- もちろん、IPアドレスやホスト名を指定すれば別のマシンと接続することができます。

```
object Form3: TForm3
object SQLConnection1: TSQLConnection
   ConnectionName = 'TDSServerModule1'
   DriverName = 'Datasnap'
   Params.Strings = (
     'HostName=127.0.0.1
     'Port=211')
   LoginPrompt = False
   Connected = True
 end
 object DSProviderConnection1: TDSProviderConnection
   ServerClassName = 'TDSServerModule1'
   Connected = True
   SQLConnection = SQLConnection1
 end
 object ClientDataSet1: TClientDataSet
   Active = True
   ProviderName = 'DataSetProvider1'
   RemoteServer = DSProviderConnection1
 end
 object DataSource1: TDataSource
   DataSet = ClientDataSet1
 end
 object DBGrid1: TDBGrid
   DataSource = DataSource1
 end
end
```

// localhostでもよい

- 設計時にDBGrid内にデータを表示させる為にはサーバーアプリを起動したまま開発しましょう。

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

D2009での開発メリット 新機能紹介 サーバーメソッド

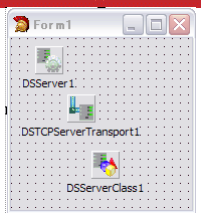
Delphi2009から

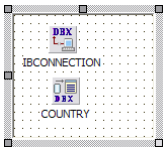
EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

- 新機能としては、以前からあったDataSnap技術で使われていたTRemoteDataModule はCOMベースの技術であったがTDSServerModuleはCOMと切り離されています。
- 接続する為に使われるSQLConnectionにDatasnapドライバが組み込まれますが、これは100% Object Pascalで記述 。exeに含まれてしまうのでクライアントへの配布はとても楽です。
- サーバーメソッドが使えます。

DataSnapサーバーアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP





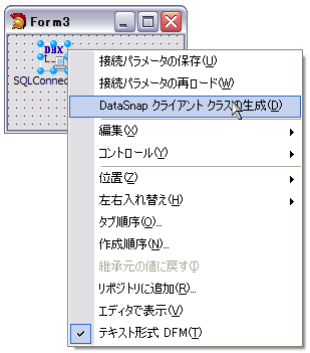
- 同じようにサーバーアプリケーションを作ります。
- そして今度は、2つ、サーバーメソッドを追加してみます。

```
TDSServerModule1 = class(TDSServerModule)
  public
    function Hello: String;
    function GetCountryDataSet: TDataSet;
  end;

function TDSServerModule1.Hello: String;
begin
  Result := 'Hello';
  ShowMessage('Hello Server World');
end;
function
   TDSServerModule1.GetCountryDataSet:
   TDataSet;
begin
  COUNTRY.Open;
  Result := COUNTRY;
end;
```

DataSnapクライアントアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP




```
object Form3: TForm3
  object SQLConnection1: TSQLConnection
    ConnectionName = 'TDSServerModule1'
    DriverName = 'Datasnap'
    LoginPrompt = False
    Connected = True
  end
end
```

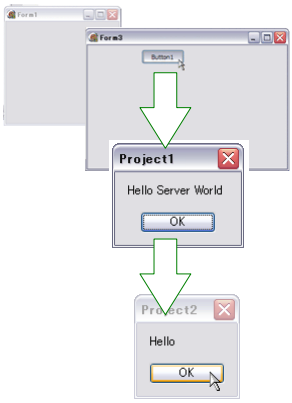
- DataSnapクライアントクラスの生成を行う。

```
//
// DataSnap プロキシ ジェネレータにより作成。
//
unit Unit4;
interface
以下、略
```

- **クライアントクラスユニット**が作成される。
  ここではunit4とする。

DataSnapクライアントアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- クライアント側でボタンイベントを記述する

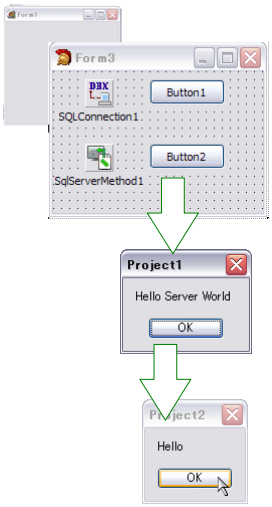
```
procedure TForm3.Button1Click(Sender: TObject);
var
  demo: TDSServerModule1Client;
begin
  Demo := TDSServerModule1Client.Create(SQLConnection1.DBXConnection);
  try
  ShowMessage(Demo.Hello);
  finally Demo.Free; end;
end;
```

TDSServerModule1Clientは、Unit4で作成されているクラス

- サーバーで実装されたメソッドが呼び出されて
- そのあと、クライアントのメソッドが呼び出されている。
- 表示されている文字列の受け渡しも可能

DataSnapクライアントアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- SQLServerMethodコンポーネントを使うと

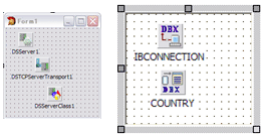
```
object SqlServerMethod1: TSqlServerMethod
  SQLConnection = SQLConnection1
  ServerMethodName = 'TDSServerModule1.Hello'
end

procedure TForm3.Button2Click(Sender: TObject);
begin
  SqlServerMethod1.ExecuteMethod;
  ShowMessage(
    SqlServerMethod1.Params[0].AsString);
end;
```

- 同じくサーバーメソッドを呼び出す事ができる。

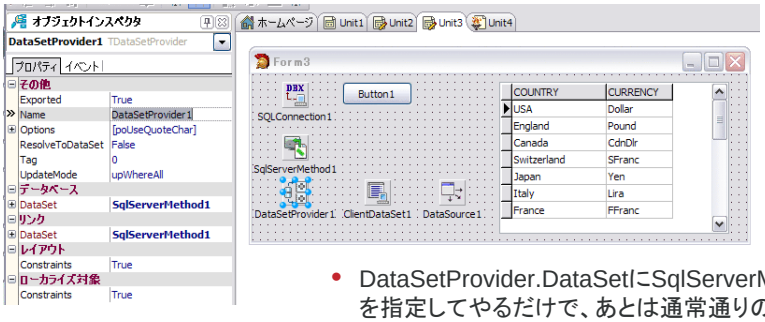
DataSnapクライアントアプリケーション作成

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- SQLServerMethodコンポーネントは戻り値としてSQLDataSetを返す事もできる

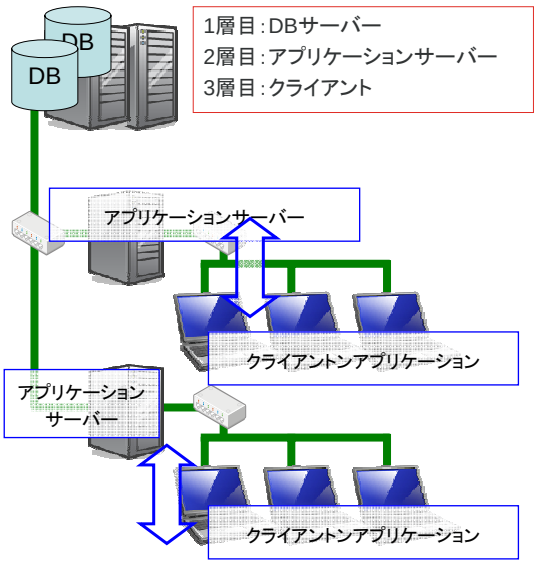
```
object SqlServerMethod1: TSqlServerMethod
  SQLConnection = SQLConnection1
  ServerMethodName = 'TDSServerModule1.GetCountryDataSet'
end
```



- DataSetProvider.DataSetにSqlServerMethodを指定してやるだけで、あとは通常通りのDB接続を行うと、DBGridにデータが表示される

3層C/Sアプリケーションによって解決される事

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP



- 見ていただいたように、DataSnap技術はサーバーとクライアントを簡単に連携させることができる技術です。
- システムの負荷分散が自在に制御することができ、それが容易です。
- ネットワーク全体を考えた構成で、システムをくみ上げていきましょう。
- **2層システムを作ったあとに同一コードを利用してすぐに3層システムへの変換もできます。**

参考情報

EMBARCADERO TECHNOLOGIES
DEVELOPER CAMP

- 参考にしました。
  - CodeGearWhitePaper DataSnap 2009 概要
    http://dn.codegear.com/print/38683
  - CodeGear公式Borland Database EngineアプリケーションのdbExpressへの移行
    http://dn.codegear.com/jp/article/33547
  - D2009 ビデオ- DataSnap多層／シンクライアントデータベースソリューションとデータリモーティング
    http://dn.codegear.com/jp/article/38595
  - Delphi Acid Floor TClientDataSetを使う
    http://www.wwlnk.com/boheme/delphi/dbtips/css0250.htm
  - Moriq Delphi D6 3層へ移行
    http://www.moriq.com/delphi/dcom.html
  - RadStudio2007 HELP アプリケーションサーバーの構造
    http://docs.codegear.com/docs/radstudio/radstudio2007/RS2007_helpupdates/HUpdate4/JA/html/devwin32/multithestructureoftheapplicationserver_xml.html
  - 今更ながら BDE (Borland Database Engine)
    http://homepage1.nifty.com/ht_deko/tech024.html#tech053
- ご協力いただいた方
  - DreamHive山本様
  - DEKO様

- 本日はありがとうございました。
- 皆さんも楽しく楽に開発できるDelphi2009を使っていきましょう。