

【セッションNo. 3】

モバイルからクラウドサービスまで！ 活用が広がる多層アプリ開発

株式会社ミガロ。
RAD事業部 技術支援課
吉原 泰介

はじめに

はじめに

モバイルアプリケーションの業務向け開発・活用も増えてきましたが、モバイルからIBM i に接続する際に重要になるのが中間サーバを使った多層構成の仕組みです。

中間サーバのアプリケーション構築手法には、従来の DataSnap Server の手法に加えて、Delphi/400 10.2 Tokyo から新しく RAD Server が使えるようになっています。

(開発ライセンスに1サイトライセンスが付属します)

本セッションでは中間サーバを使った多層アプリケーションをテーマに、RAD Server を使った開発手順や新技術を使った拡張などの情報をご紹介します。

【アジェンダ】

1. 多層アプリケーションとは？
2. 中間サーバのアプリケーションを開発する手法
3. RAD Serverを使った実装手順
4. Enterprise Connectorsを利用した連携
5. まとめ

補足資料

- ・ DataSnapServerを使った実装手順 (FireDAC)
- ・ DataSnapServerを使った実装手順 (dbExpress)
- ・ 他言語アプリケーションへの連携

1. 多層アプリケーションとは？

■ 多層アプリケーションとは？

- 多層アプリケーションとは、複数層で構成されたアプリケーションのこと。

例えばClient/Serverアプリケーションは、クライアント層とデータ層で処理される2層構成、WebアプリケーションやモバイルアプリケーションでIBM i にアクセスする場合は、中間層のサーバを経由する3層構成となっている。



■ 多層アプリケーションとは？

- よく使われるアプリケーションの構成

【2層アプリケーションの構成】

Client/Serverアプリケーション



クライアント



IBM i

【3層アプリケーションの構成】

Webアプリケーション



クライアント(ブラウザ)



Webサーバ



IBM i

モバイルアプリケーション



クライアント



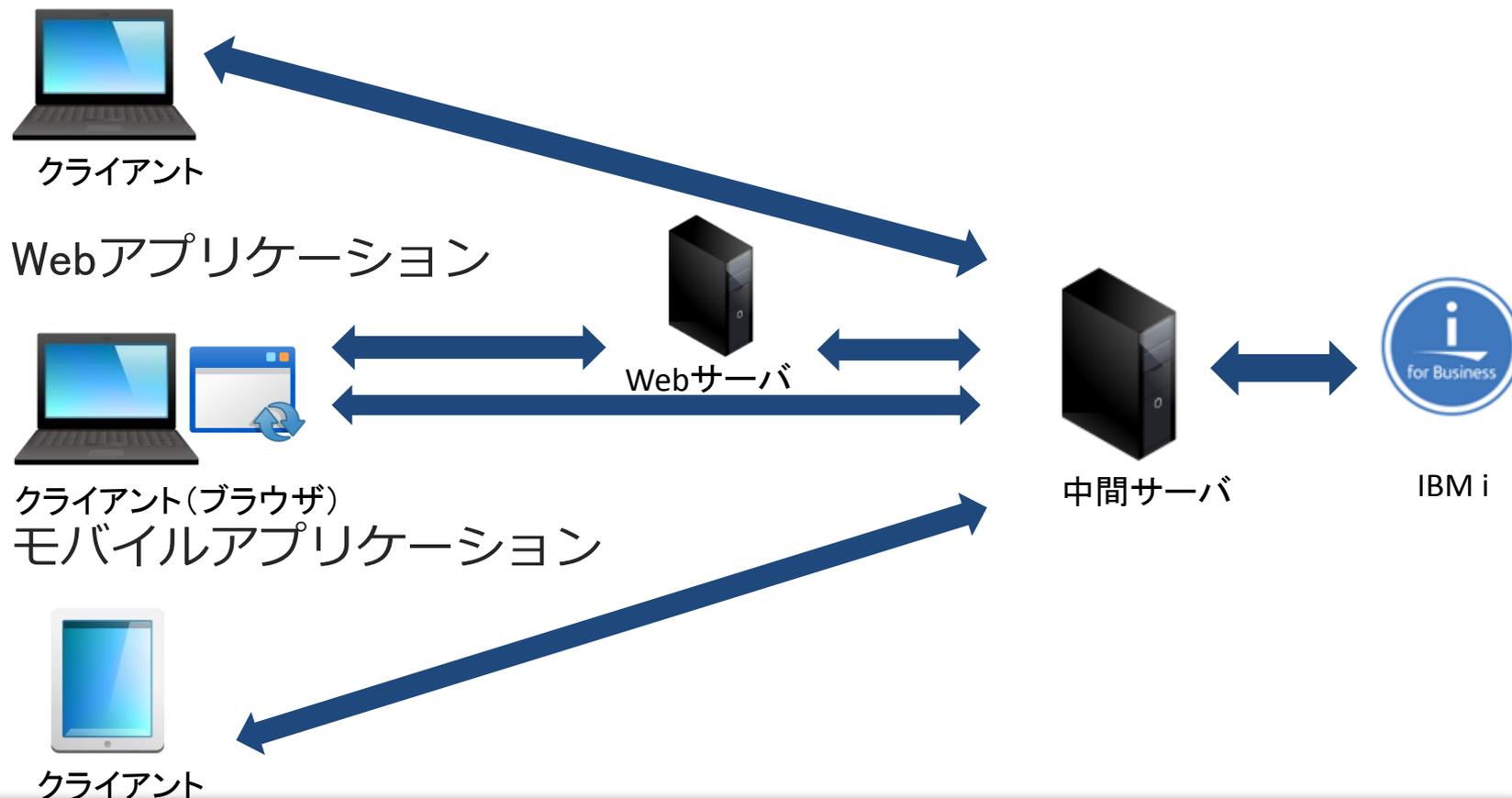
中間サーバ



IBM i

■ 多層アプリケーションとは？

- 中間サーバを活用したアプリケーション構成
【中間サーバを活用したアプリケーションの構成例】
Client/Serverアプリケーション



■ 中間サーバを経由する構成のメリット

- メリット①

ビジネスロジックの分離

画面などのGUI部分とデータ側の業務処理を含むビジネスロジック部分を分離して開発・メンテナンスすることができる。

これによりアプリケーションの実装は目的別のシンプルな内容になる。

単一EXEのアプリケーション

GUI

ビジネス
ロジック

DX

2層のクライアント

クライアントアプリケーション 中間サーバアプリケーション

GUI

ビジネス
ロジック

DX

3層のクライアント

分離

DX

中間サーバ

■ 中間サーバを経由する構成のメリット

• メリット②

クライアントアプリケーションのダウンサイズ・配布の軽減
ユーザー環境へ配布するクライアントアプリケーションはビジネスロジックを含まないため、実行ファイルのサイズを小さく抑えることができ、変更や再配布の頻度も減らすことができる。
(ビジネスロジック側の変更ではクライアントアプリケーションは変わらない)
また使用するDB関連の導入や設定等もクライアント側では不要になる。

クライアントモジュール
は導入してる？

バージョンは？
環境設定は？



2層のクライアント



DB関連は中間サーバ側で
導入・設定するので不要



3層のクライアント

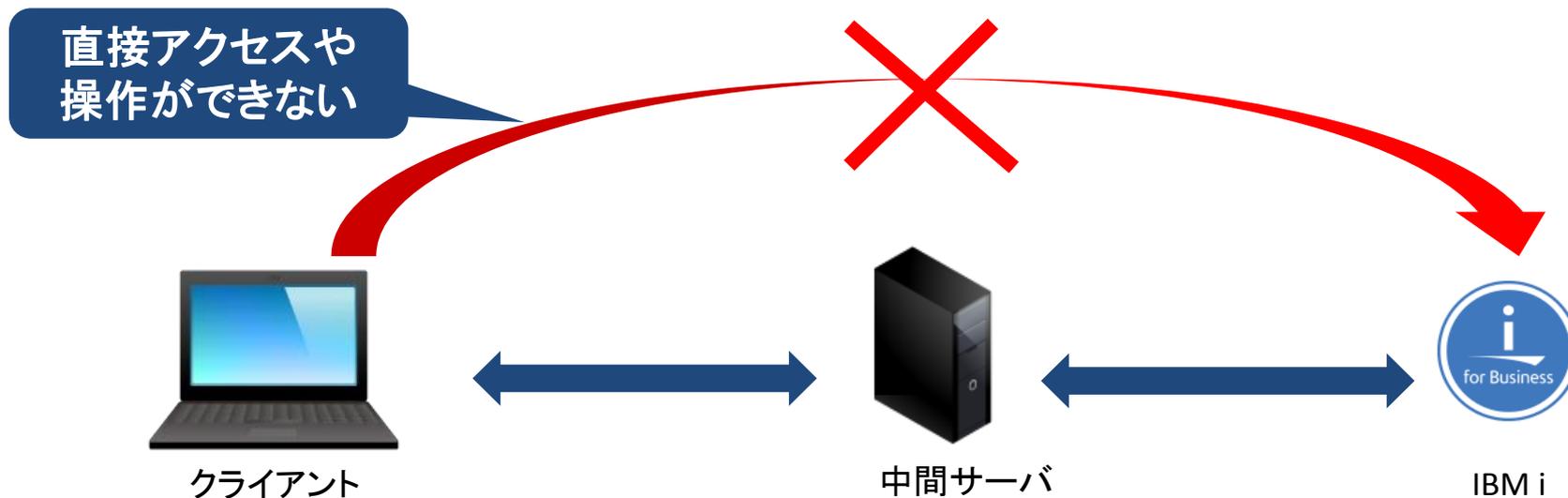


■ 中間サーバを経由する構成のメリット

- メリット③

セキュリティ

クライアント環境からDBに直接アクセスできない構成で環境を構築できる。特に外部からも利用する環境を考える場合には、中間サーバの環境でセキュリティ上の配慮・対策がしやすい。またデータ通信もHTTPS等による暗号化データでやりとりも可能。



■ 中間サーバを経由する構成のメリット

• メリット④

マルチデバイス対応・展開

ビジネスロジックを中間サーバ側のアプリケーションで持つことで、クライアントアプリケーションがデバイス毎に異なっても共通で対応できる。（同じビジネスロジックをデバイス毎に作成不要）

Delphi/400でマルチデバイス開発したアプリケーションであればワンソースでの管理もでき、中間サーバ側との親和性も高い。



Delphi/400で開発したマルチデバイスアプリは中間サーバアプリと親和性が高い

■ 中間サーバを経由する構成の考慮点

- 考慮点

多層構成では IBM i への接続は中間サーバからとなる為、
接続ユーザーは処理実行時のみジョブを利用する方式になる。
(C/Sアプリケーションのようにジョブが継続するわけではない)

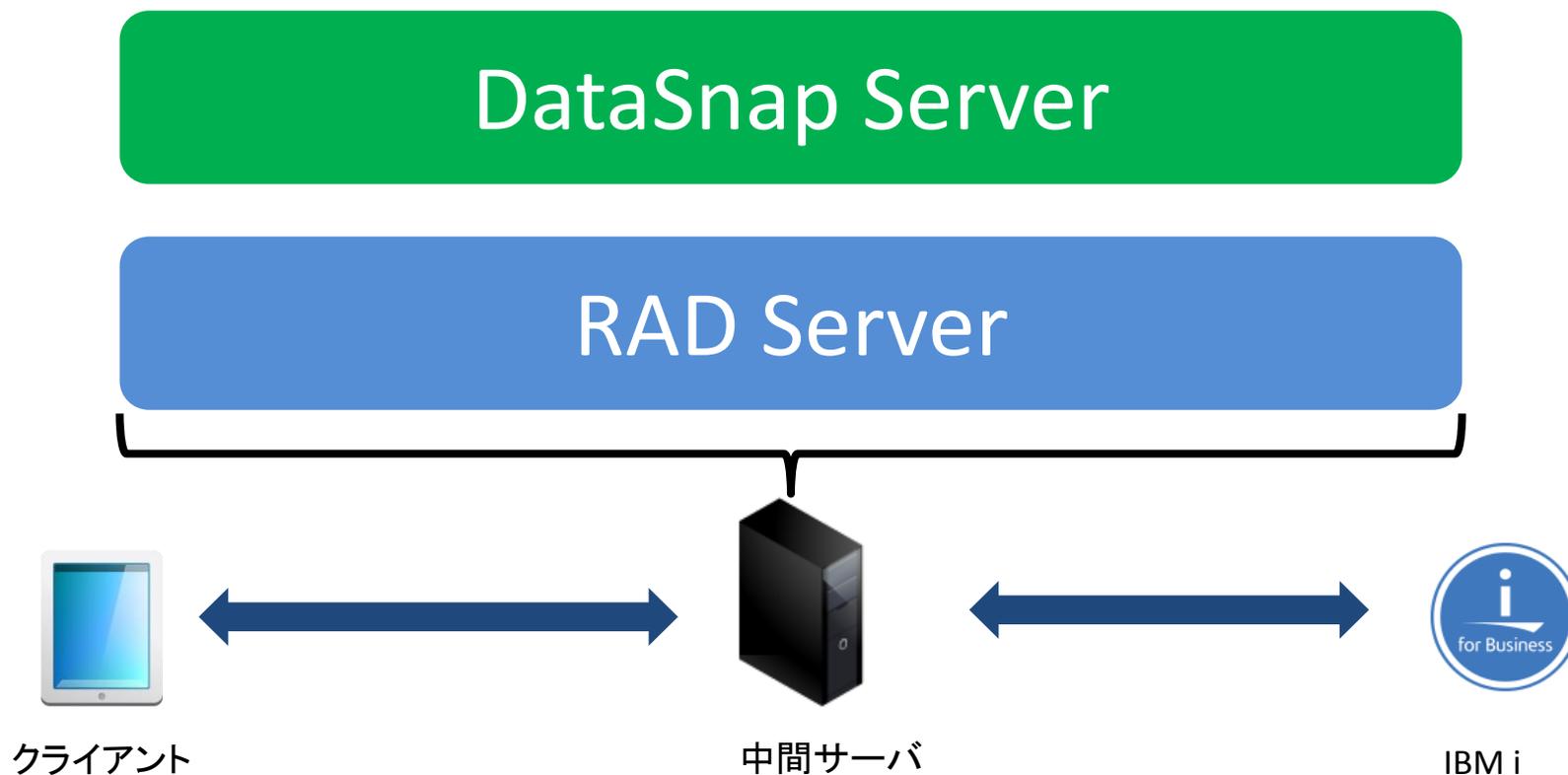
QTEMPなどジョブに依存した仕組みは不向き

メンバを使った仕組みは向いている

2. 中間サーバのアプリケーションを開発する手法

■ 中間サーバのアプリケーションを開発する手法

- Delphi/400を使って中間サーバ側のアプリケーションを開発する手法はいくつかありますが、主な2手法をご紹介します



■ DataSnap Serverとは？

• 特徴

- 多層アプリケーションの開発を可能にするSDK
- サーバ機能はプログラムで開発する必要がある
- 開発での実装となるため、プログラムの自由度が高い
- TCP/IP、HTTP (S)、REST、JSON、COMなどの標準技術をサポート

■ RAD Serverとは？

- 特徴

- 多層アプリケーションのREST APIを公開するサーバ
- サーバで必要となる高度な機能がいくつも提供されている
- ユーザー管理機能、認証機能、分析機能などの標準機能を豊富に搭載
- HTTP (S)、 REST、 JSONなどの標準技術をサポート

■ DataSnap ServerとRAD Serverの主な違い

	DataSnap Server	RAD Server
機能開発	全て開発で実装が必要	必要な部分のみ開発
標準通信	TCPIP/HTTP(S)	HTTP(S)
DBエンジン	FireDAC、dbExpress	FireDAC
モバイル対応機能	開発が必要	Push通知、デバイス認証等が標準機能
管理ツール	開発が必要	標準で付属(分析も可能)
ライセンス	開発ライセンスに含まれる (Enterprise以上)	開発ライセンスに 1サイトライセンス付属 (10. 2 Tokyo Enterprise以降)

■ 実装手順の違い

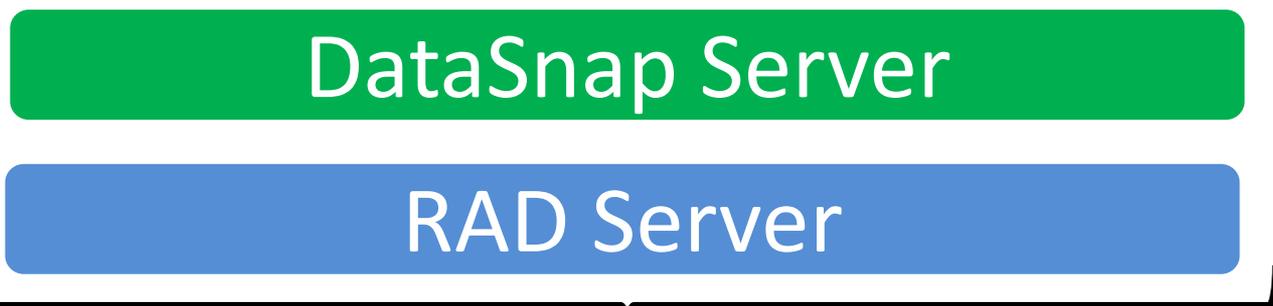
- DataSnap ServerとRAD Serverでは中間サーバで担う役割は似ているが、実装内容や手順は異なる。

新機能RAD Serverについて、実装手順を簡単なデータアクセスを題材にご説明

(従来のDataSnap Server実装手順は補足資料P59～を参考ください)



クライアント



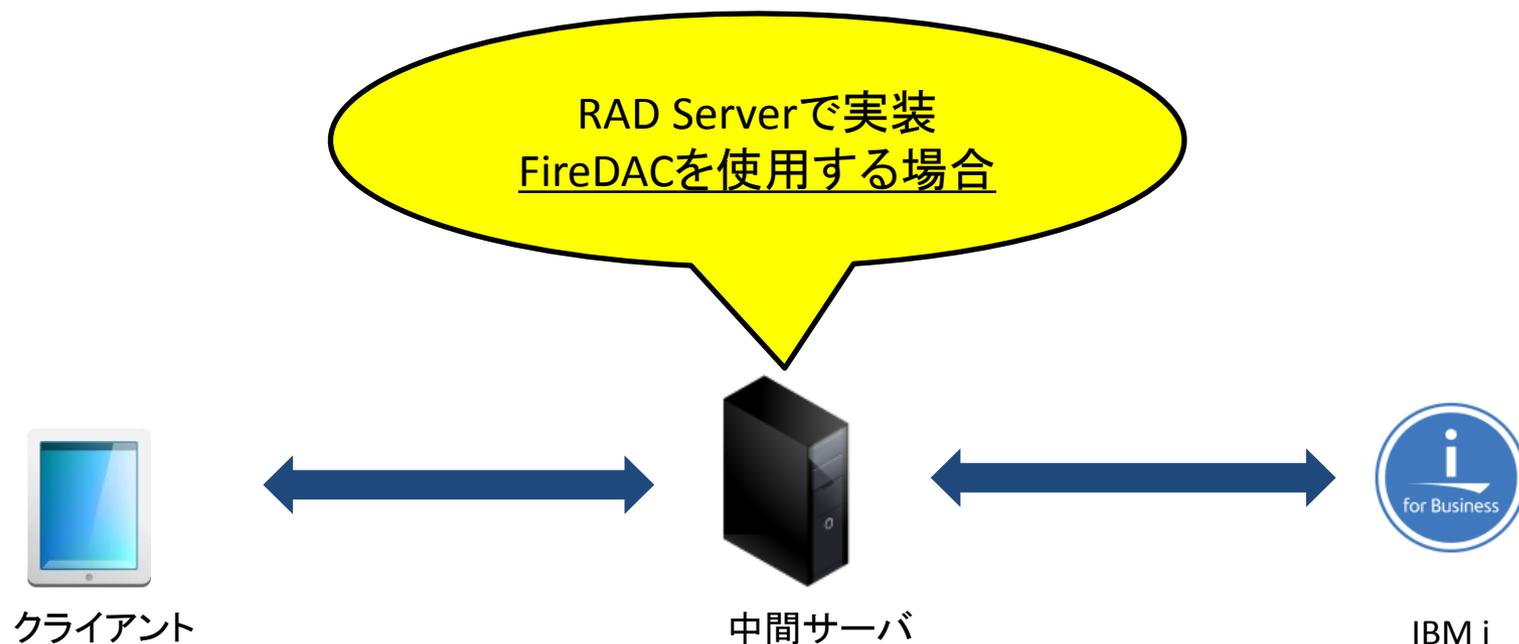
中間サーバ



IBM i

3. RAD Serverを使った実装手順

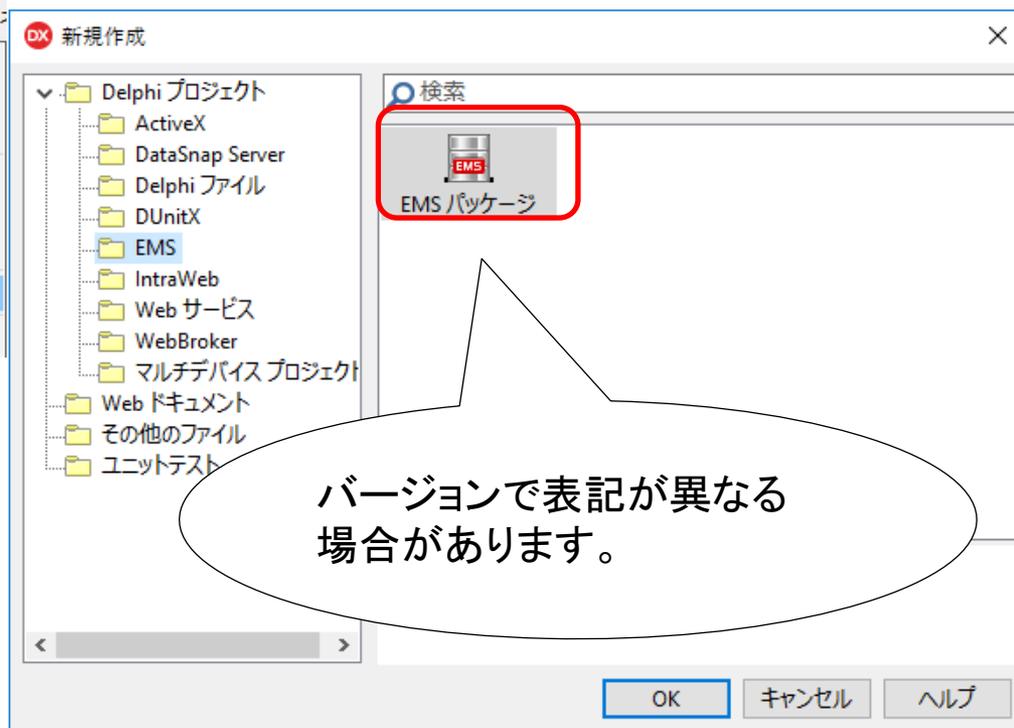
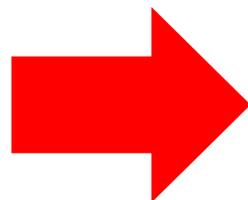
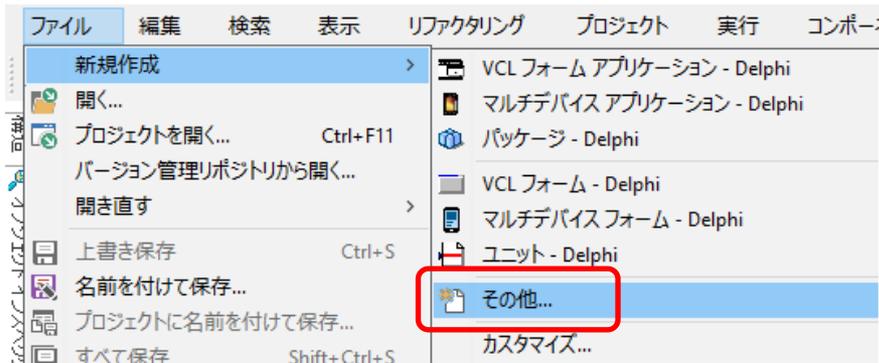
■ RAD Serverを使った実装手順



■ RAD Serverを使った実装手順

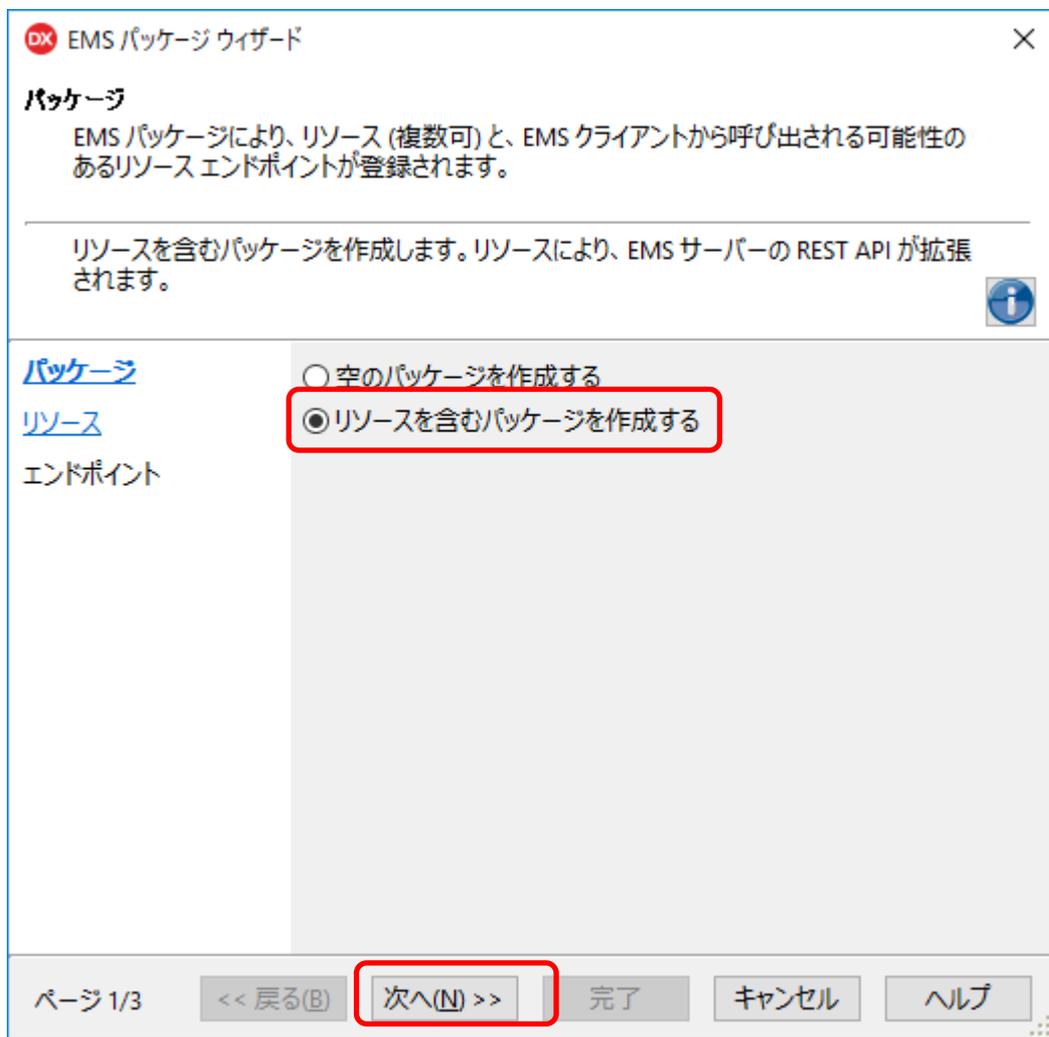
- 実装手順1
プロジェクトの作成

DX RAD Studio 10.2



■ RAD Serverを使った実装手順

- 実装手順2
ウィザードの指定



■ RAD Serverを使った実装手順

- 実装手順3
ウィザードの指定

DX EMS パッケージウィザード

リソース
リソースを追加すると、EMS サーバーの REST API を拡張できます。

リソースデータモジュールを作成します。リソースデータモジュールは、デザイナとコンポーネントを使ってリソースを実装するためのものです。

パッケージ
リソース
エンドポイント

リソース名(R): CUST

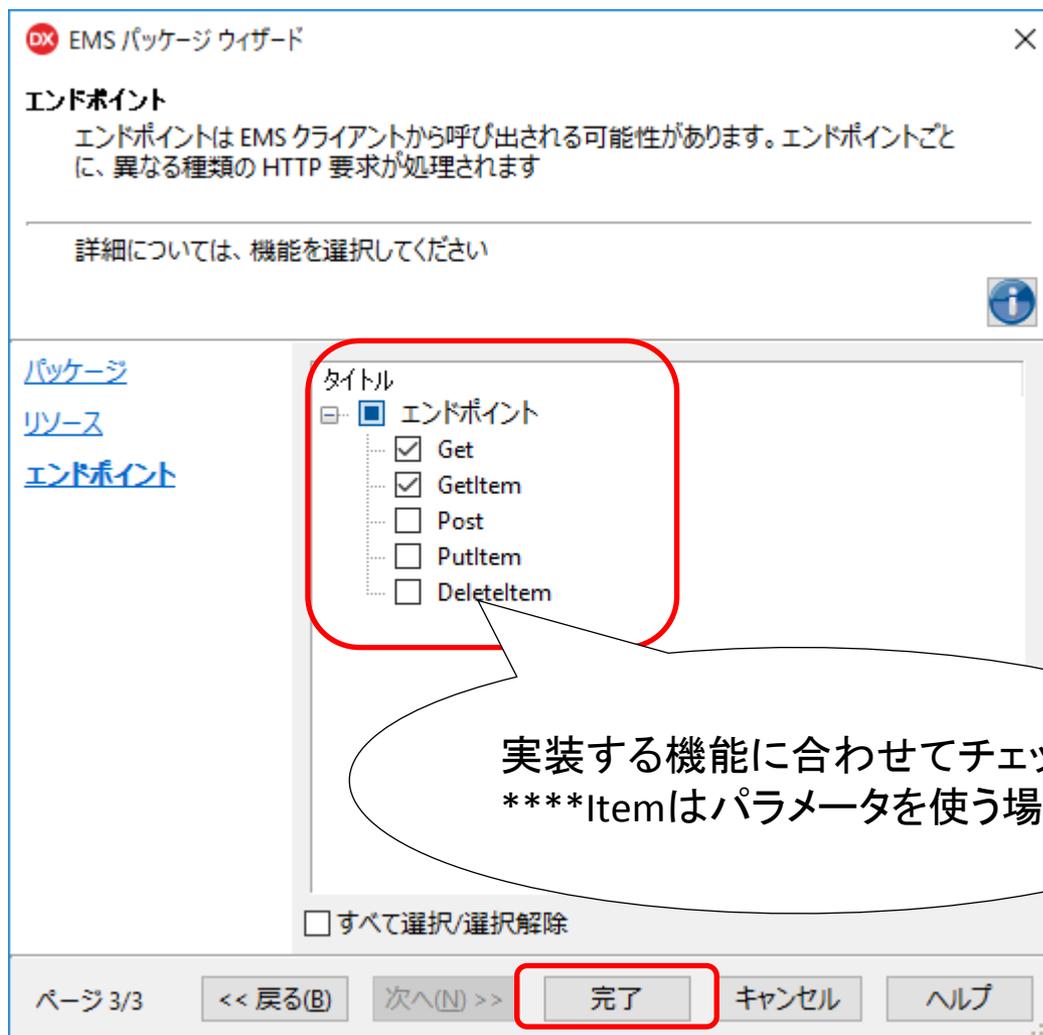
ファイルの種類(T): データモジュール
 ユニット

RESTサービスとして呼び出す際は、このリソース名を使用

ページ 2/3 << 戻る(B) 次へ(N) >> 完了 キャンセル ヘルプ

■ RAD Serverを使った実装手順

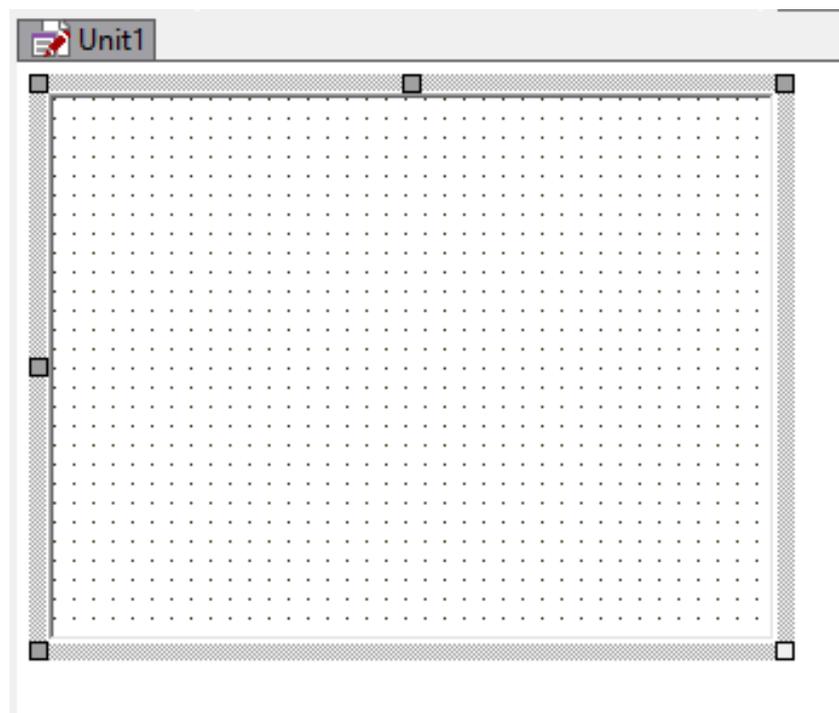
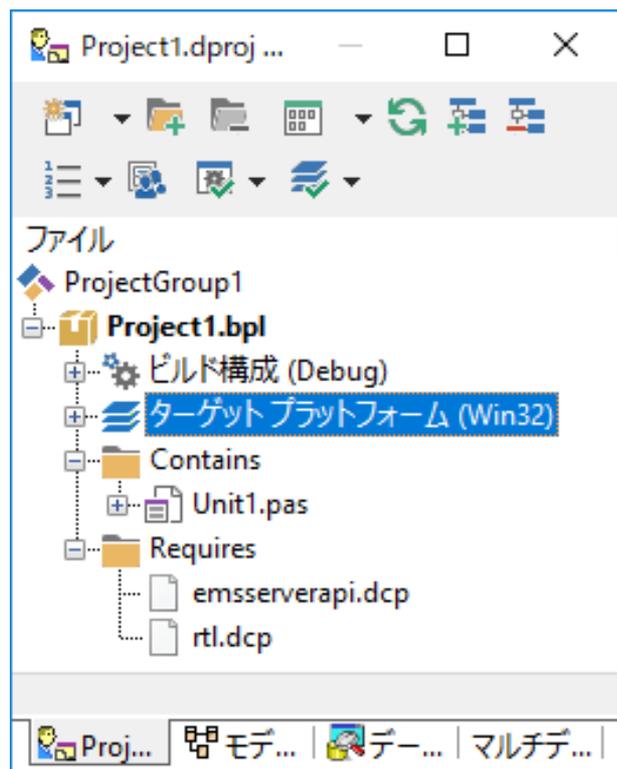
• 実装手順4 ウィザードの指定



■ RAD Serverを使った実装手順

• 実装手順5

自動生成されるユニット

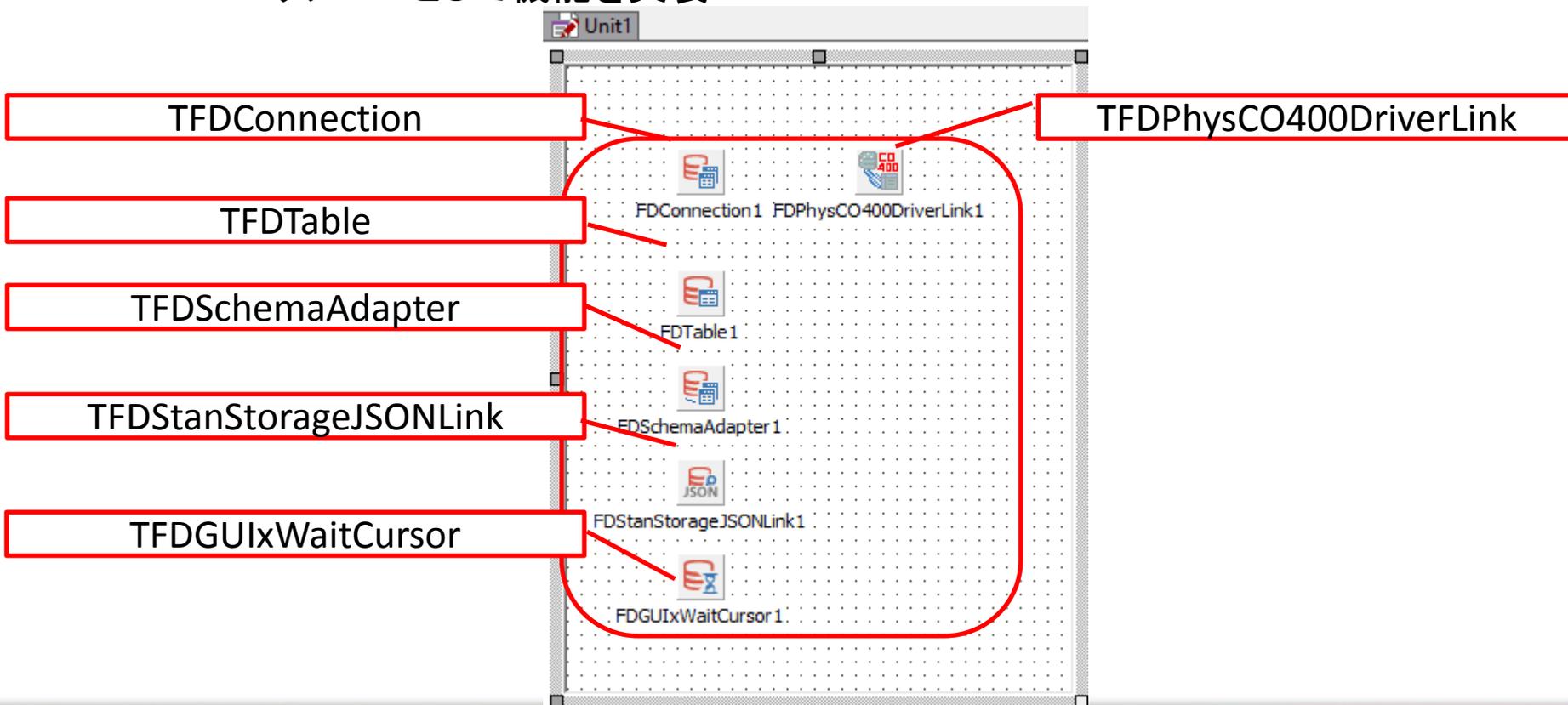


■ RAD Serverを使った実装手順

• 実装手順6

サーバ機能のコンポーネント配置

リソースとして機能を実装



■ RAD Serverを使った実装手順

- 実装手順7
コンポーネントの設定

TFDConnection

FireDAC 接続エディタ - [FDConnection1]

ドライバまたはオーバーライドする接続定義の名前を選択してから、パラメータをセットアップします

定義 オプション 情報 SQL スクリプト

ドライバ ID(D): CO400

接続定義名(N):

テスト(T) ウィザード(W) デフォルトに戻す(R) ヘルプ(H)

パラメータ	値	デフォルト
Pooled	False	False
Database	POWER8	
User_Name	D4TEC	
Password	D4TEC	
MonitorBy		
ODBCAdvanced	LibraryOption=D4TEC22LIB	
LoginTimeout		
Alias		
Server		
Port		
ExtendedMetadata		
MetaDefSchema		
MetaCurSchema		

キャンセル(C)

通常のIBM i 接続設定

TFDTable

オブジェクト インспекタ

FDTable1 TFDTable

検索

プロパティ イベント

ResourceOptions	(TFDBottomResourceOption)
SchemaAdapter	FDSchemaAdapter1
SchemaName	
TableName	CUSTOMER
Tag	0
Transaction	
IUpdateObject	

フィールドエディタ... ビジュアルにバインド...
バインドソースの追加 クイック編集...

すべての項目が表示されています

■ RAD Serverを使った実装手順

• 実装手順8

データ取得の機能を実装

```
procedure TCUSTResource1.Get(const AContext: TEndpointContext; const
ARequest: TEndpointRequest; const AResponse: TEndpointResponse);
var
  oStr: TMemoryStream;
begin
  oStr := TMemoryStream.Create;

  // クエリの実行結果をスキーマアダプタから
  // メモリストリーム経由で返す
  FDTTable1.Open;
  FDSchemaAdapter1.SaveToStream(oStr,TFDStorageFormat.sfJSON);
  AResponse.Body.SetStream(oStr,'application/json', True);
end;
```

RESTでパラメータを受け取ることも可能
ARequest.Params.Values['XXXX']
※QueryのSQLでバインド変数として
Param値で利用すると便利

■ RAD Serverを使った実装手順

• 実装手順9

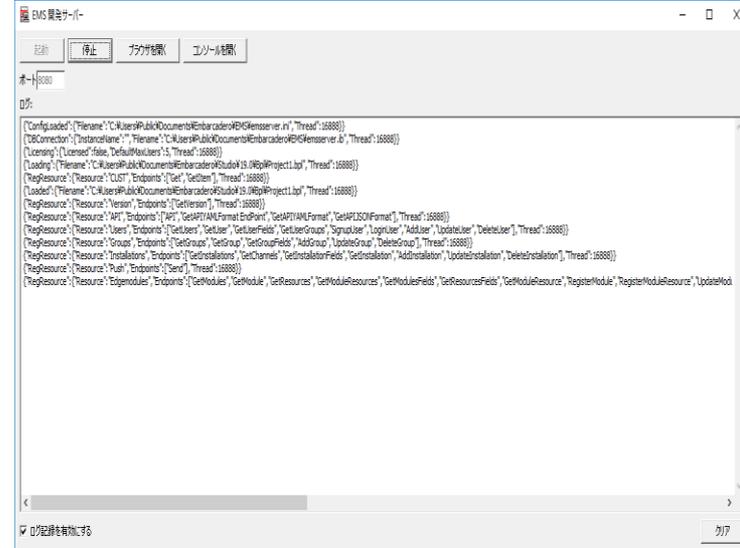
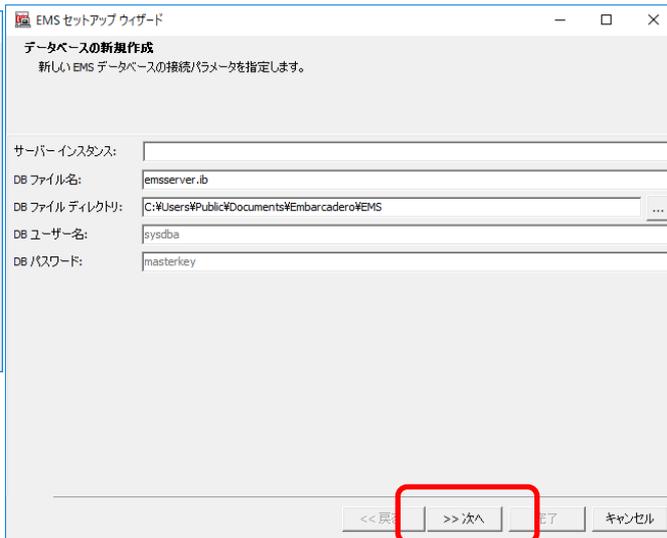
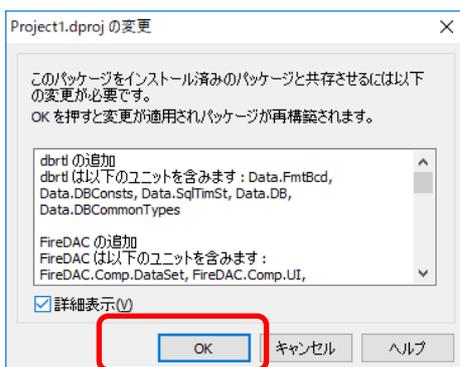
コンパイル・実行

初回コンパイルや実行時はInterBaseの設定も含め、ダイアログがいくつか出るので応答が必要。

パッケージの参照

InterBaseの設定
(サーバ自体の管理データ)

起動画面



■ RAD Serverを使った実装手順

RAD Serverで実装
FireDACを使用する場合



クライアント



中間サーバ

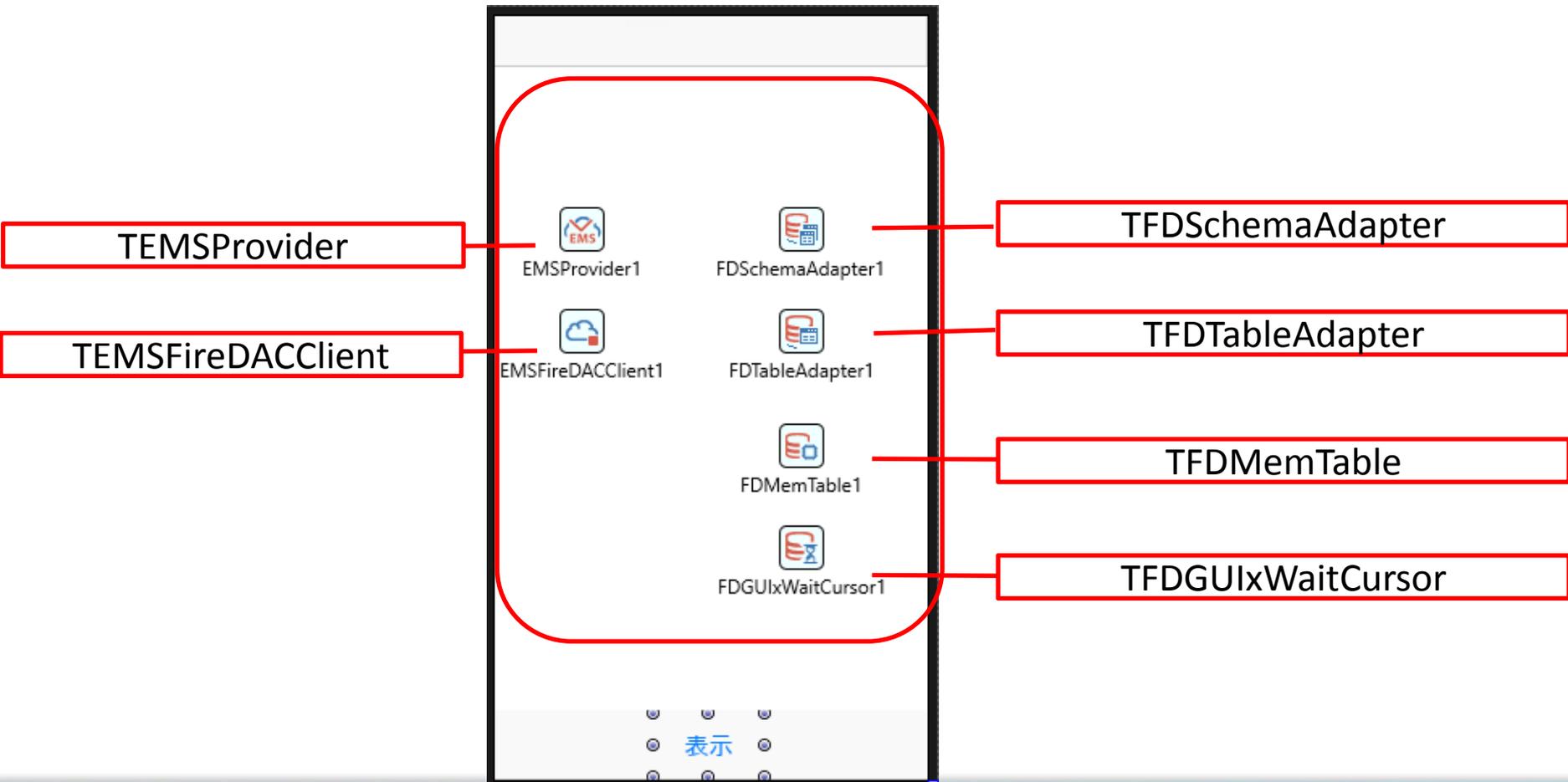


IBM i

■ RAD Serverを使った実装手順

• 実装手順10

クライアントアプリケーションのコンポーネント配置



■ RAD Serverを使った実装手順

- 実装手順11
コンポーネントの設定

TEMSPProvider

オブジェクトインスペクタ

EMSPProvider1 TEMSPProvider

検索

プロパティ イベント

ProxyPort	0
ProxyServer	
URLHost	999.999.999.999
URLPort	8080
URLProtocol	http

サーバIPやポートを設定。

接続テスト クイック編集...

すべての項目が表示されています

TEMSPFireDACClient

オブジェクトインスペクタ

EMSPFireDACClient1 TEMSPFireDACClient

検索

プロパティ イベント

Auth	
LiveBinding デザイン	LiveBinding デザイン
Name	EMSPFireDACClient1
Provider	EMSPProvider1
Resource	CUST
SchemaAdapter	FDSchemaAdapter1
Tag	

接続したサーバで使用されるリソースなどを指定。

クイック編集...

すべての項目が表示されています

TFDTableAdapter

オブジェクトインスペクタ

FDTableAdapter1 TFDTableAdapter

検索

プロパティ イベント

ColumnMappings	TFDDAptColumnMappings
DatTableName	FDTTable1
DeleteCommand	
FetchRowCommand	
InsertCommand	
LiveBinding デザイン	LiveBinding デザイン
LockCommand	
MetalInfoMergeMode	mmReset
Name	FDTableAdapter1
SchemaAdapter	FDSchemaAdapter1

サーバ側のデータセット

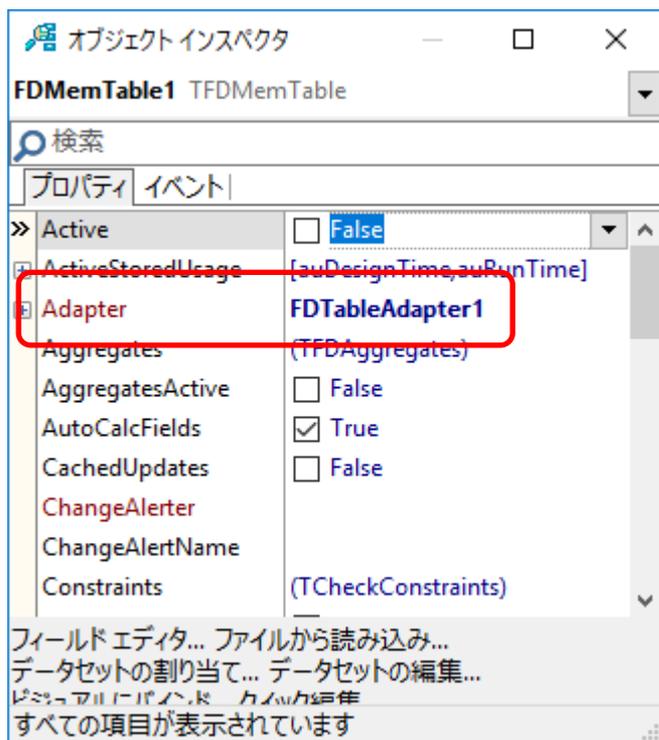
クイック編集...

すべての項目が表示されています

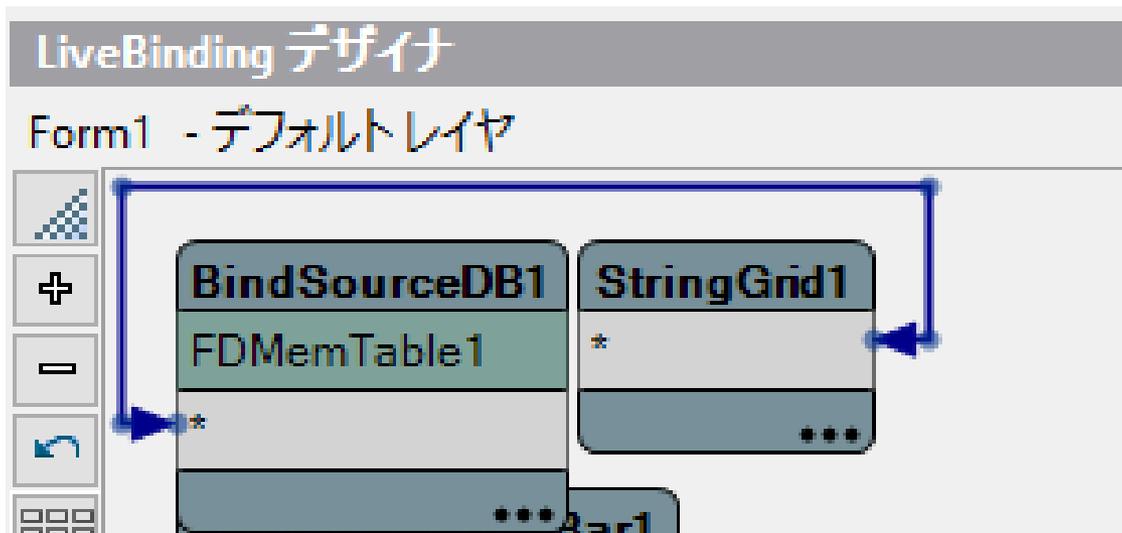
■ RAD Serverを使った実装手順

- 実装手順12
コンポーネントの設定

TFDMemTable



ライブバインディング設定



■ RAD Serverを使った実装手順

- 実装手順13

データ取得の機能を実装

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    //設定しているリソースのGetDataを呼び出す  
    EMSFireDACClient1.GetData;  
end;
```

■ RAD Serverを使った実装手順

- 実行

ID	NAME
1221	ココナッツマリンショップ2
1513	ダイブハウスタートル5
3444	ダイビングベース新井8
1231	アクアダイビングセンター
1351	亀山ダイブセンター
1380	ダイブショップブルーリーフ
1384	MHMダイバーズクラブ
1984	ADVENTURE UNDERSEA
2118	グリーンスポーツクラブ
2135	バイナッブルダイバーズ
2156	マリンハウスペンぎん
2163	SCUBA HEAVEN
2165	SHANGRI-LA SPORTS CENTER
2355	ダイブショップマーメイド
2975	ダイブリゾートあしか
2984	サンセットダイビングサービス
3041	マリンショップアクア



デスクトップでも
モバイルでも同じソースで
実装可能

クライアントアプリ



中間サーバ



IBM i

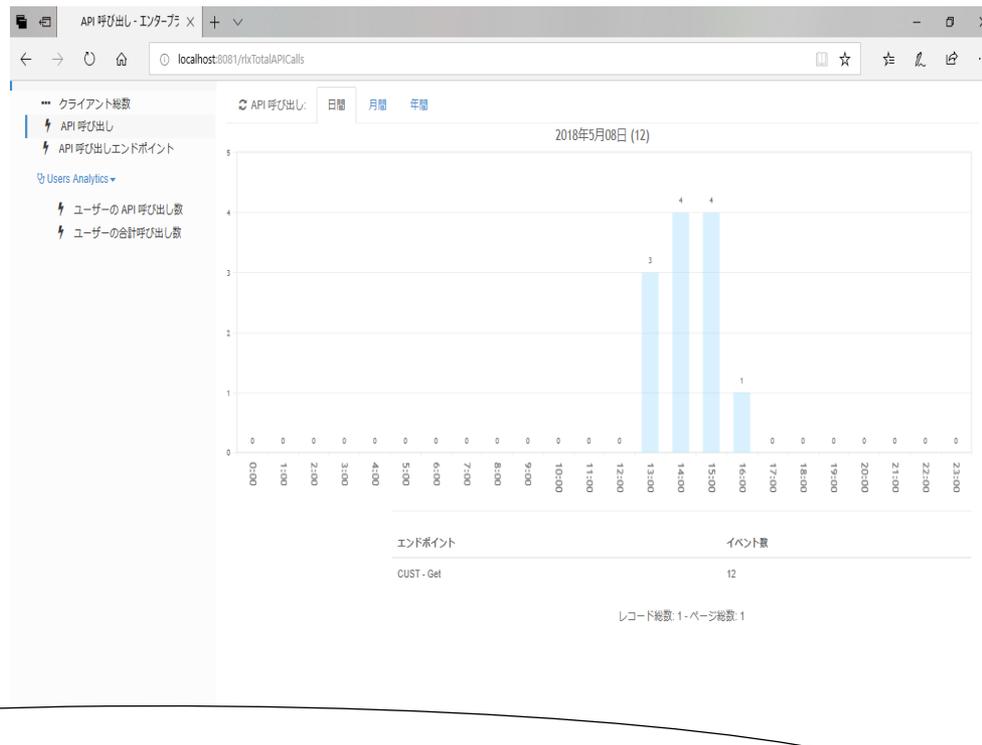
■ RAD Serverのコンソール

- コンソールのログイン
ログインアカウント情報はiniファイルに 設定されています。
C:¥Users¥Public¥Documents¥Embarcadero¥EMS¥emsserver.ini



■ RAD Serverのコンソール

- コンソールで使える分析メニュー（アナリティクス）



APIが利用された回数や時間など一目瞭然
ユーザーが実際に使っている機能や頻度が把握できる！

4. Enterprise Connectorsを利用した連携

■ Enterprise Connectorsを利用した連携

- Enterprise Connectorsとは？

Delphi/400からsalesforceやAWS、SAP、ERP、Office 365、Googleドキュメント、ビッグデータDB、決済、ソーシャルサービスなど、80種類を超えるクラウドデータやエンタープライズサービスへのアクセス機能を追加できるパッケージソリューション



従来のDBアプリ開発スキルで簡単に実装が可能

Delphi/400に標準搭載されたDBエンジン「FireDAC」に対応しており、各サービスを利用するためのAPIは、コンポーネントによって提供される。

開発者はこれまでのDBアプリケーション開発スキルでクラウドデータ等を簡単に扱うことが可能。

■ Enterprise Connectorsを利用した連携

- Enterprise Connectorsを中間サーバで利用するメリット
 - IBM i 以外に、クラウドサービスやパッケージのデータが連携できる
 - FireDACのコンポーネントで開発できるので、習得や調査が不要
 - クラウドサービスやパッケージのAPI変更にプログラム対応が不要
 - 中間サーバに実装すれば各クライアントでの導入や設定が不要

開発者の負担となる工数・コストが大幅軽減してアプリの拡張ができる

CRM & マーケティング

Salesforce.com	Microsoft Dynamics CRM (On-...	NetSuite CRM
SugarCRM	Highrise	Zoho CRM
Bullhorn CRM	Oracle Sales Cloud	Veeva
Google Analytics	Google AdWords	MailChimp
Oracle Eloqua	HubSpot	Marketo
SendGrid	YouTube	YouTube Analytics
Magento		

会計

Microsoft Dynamics GP	QuickBooks	QuickBooks Online
QuickBooks Point-Of-Sale	Sage 50	Sage 50 UK
Xero Accounting	Exact Online	Intacct
FreshBooks	Reckon	

ソーシャルネットワーク

Force.com	Twitter	Facebook
LinkedIn		

NoSQL & ビッグデータ

DynamoDB	Google BigQuery
Cassandra	1010data
HPCC Systems	MongoDB
HBase	MySQL
Active Directory	Elasticsearch

ドキュメント& ファイル

Excel Files	SharePoint Excel Services
CSV/TSV	XML

Eコマース/財務

Authorize.Net	Square
PayPal	Shopify
OFX Transactions	E*TRADE
Quandl	

ネットワーク & 認証

OData	JSON	LDAP
OFX Transactions	Email	RSS

ERP & コラボレーション

Azure Storage	Microsoft SharePoint (On-pre...	SharePoint Excel Services	Google Apps
Elasticsearch	Office 365	Basecamp	Microsoft Project
Couchbase	Smartsheet.com	Magento	Microsoft Dynamics GP
Access	Microsoft Dynamics NAV	Microsoft Dynamics AX	NetSuite ERP
xBase	SAP NetWeaver	Acumatica ERP	oDoo ERP
	ServiceNow	Splunk	Basecamp
	JIRA		

その他

eBay	Twilio	SimpleDB	Google Gmail
Stripe	PowerShell	Bing Search	Microsoft Exchange
OpenExchangeRate	Google Search		

Enterprise Connectorsを利用した連携

【n層アプリケーション】
Client/Serverアプリケーション



クライアント

Webアプリケーション



クライアント(ブラウザ)



Webサーバ



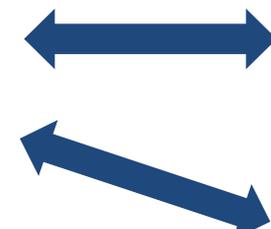
中間サーバ

モバイルアプリケーション



クライアント

Salesforce & Force.com	Microsoft Dynamics CRM	Microsoft Dynamics GP
Microsoft Dynamics NAV	Microsoft Dynamics AX	Microsoft SharePoint
NetSuite CRM & ERP	QuickBooks Desktop	QuickBooks Online
QuickBooks Point of Sale	Sage US	Sage 50 UK
Xero Accounting	Exact Online	Intacct
FreshBooks Accounting	SAP NetWeaver	OData Services
JSON Services	Microsoft Excel	SharePoint Excel Services
Twitter	Facebook	LDAP Directory Services
Amazon DynamoDB	Amazon SimpleDB	Google Analytics
Google Apps	Google Apps	Google Sheets
Office 365	Gmail	OFX Financial Accounts
PowerShell	Email	RSS Feeds
MongoDB	Google BigQuery	Azure Table
Apache Cassandra	Couchbase Server	Apache HBase



IBM i



他DB

■ Enterprise Connectorsの組み込み例 (salesforce)

● 組み込み手順1

[ツール|GetItパッケージマネージャ]から対象のECを検索して選択

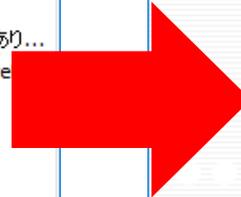
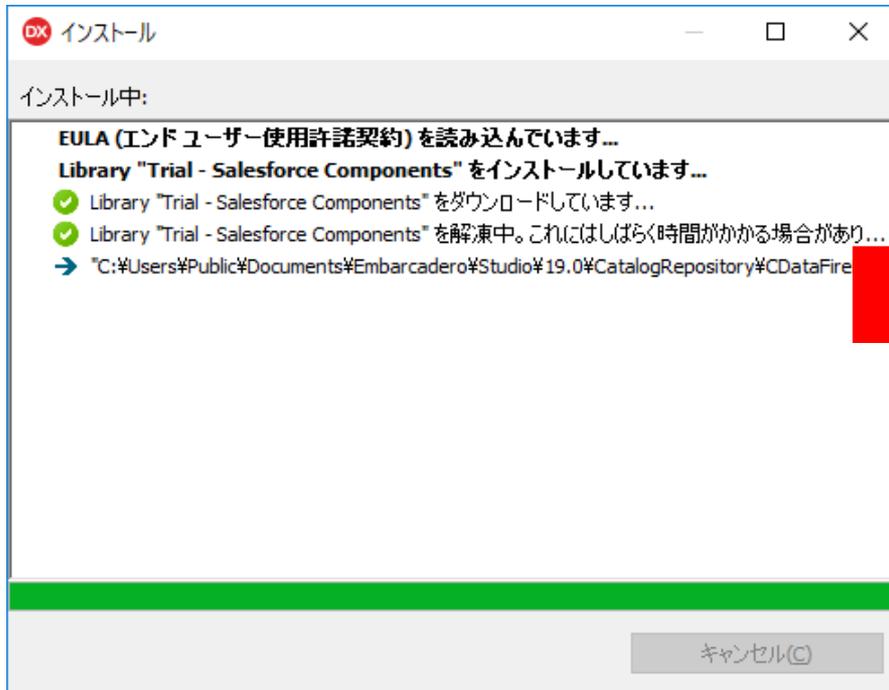
The screenshot shows the 'GetIt パッケージマネージャ' window with a search for 'salesforce'. Three connector packages are listed: 'Beta - Salesforce Einstein', 'Trial - Salesforce Components', and 'Trial - Salesforce Chatter'. A red circle highlights the 'Trial - Salesforce Components' package. A red arrow points from this package to a '依存先コンポーネントのライセンス' (License of the dependent component) window. This window displays the license agreement for 'Trial - Salesforce Connector', including a '説明' (Description) and a 'ライセンス' (License) section. The license text states: 'THIS SOFTWARE LICENSE AGREEMENT IS A LEGAL AGREEMENT BETWEEN YOU, EITHER A SINGLE INDIVIDUAL, ENTITY OR GOVERNMENT ORGANIZATION AND EMBARCADERO TECHNOLOGIES, INC. AND ITS AFFILIATES FOR THE SOFTWARE YOU ARE LICENSING. CAREFULLY READ THIS AGREEMENT BEFORE YOU INSTALL OR USE THE SOFTWARE. BY INSTALLING OR USING THE SOFTWARE OR BY CLICKING ON "ACCEPT" YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT AND YOU REPRESENT THAT YOU HAVE THE AUTHORITY TO ENTER INTO THIS AGREEMENT. ALL SOFTWARE ORDERED THROUGH AN AUTHORIZED RESELLER OR DISTRIBUTOR IS GOVERNED BY THIS AGREEMENT. IF YOU DO NOT HAVE THE AUTHORITY TO ENTER INTO THIS AGREEMENT, OR IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, YOU MUST NOT INSTALL OR USE THE SOFTWARE. THIS AGREEMENT, SHALL GOVERN YOUR INSTALLATION AND USE OF THE PRODUCTS UPON THE EARLIER OF YOUR AGREEMENT TO PURCHASE A LICENSE FOR SUCH PRODUCTS OR YOUR INSTALLATION OR USE OF THE PRODUCTS. THIS SOFTWARE LICENSE AGREEMENT (this "Agreement"), dated as of the date of your purchase or receipt of a license to use the Software, is between Embarcadero Technologies, Inc., a Delaware corporation ("Licensor") and the customer set forth on'. At the bottom of the license window, there are two buttons: 'すべて同意する(A)' (I agree to all) and '同意しない(D)' (I do not agree).

パッケージはコネクタ毎(サービス毎)にインストールが必要

※製品購入版はCData社サイトよりインストーラをDLしてインストールします

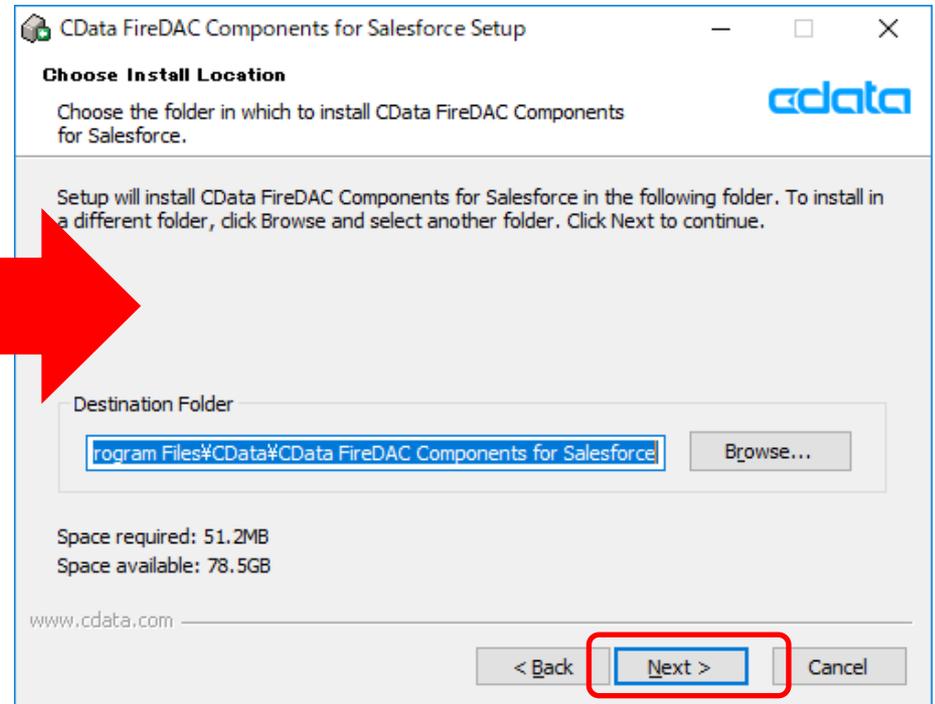
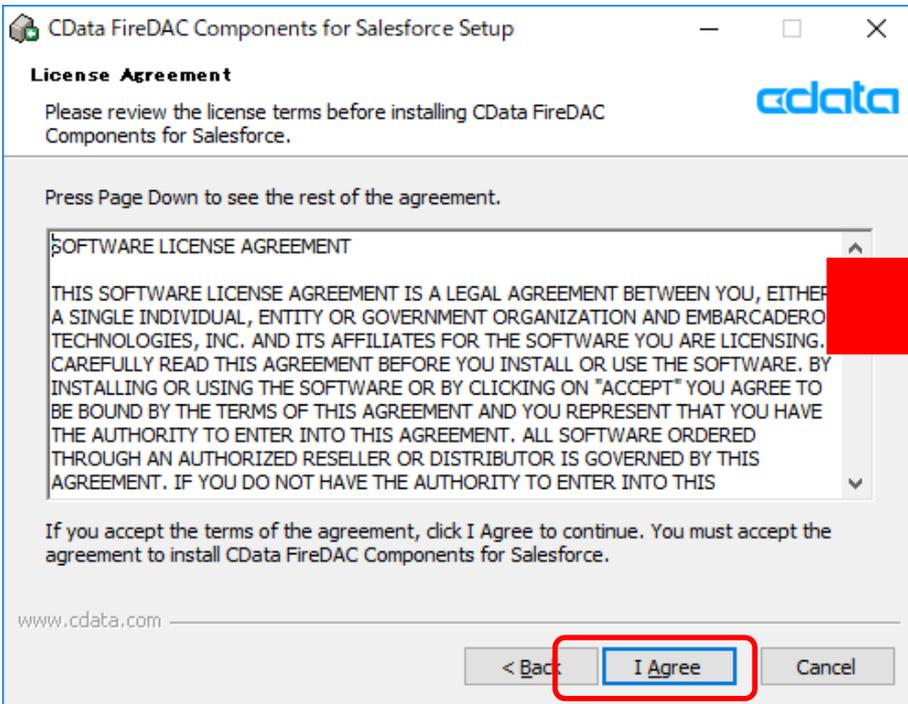
■ Enterprise Connectorsの組み込み例 (salesforce)

- 組み込み手順2
自動ダウンロード・配置



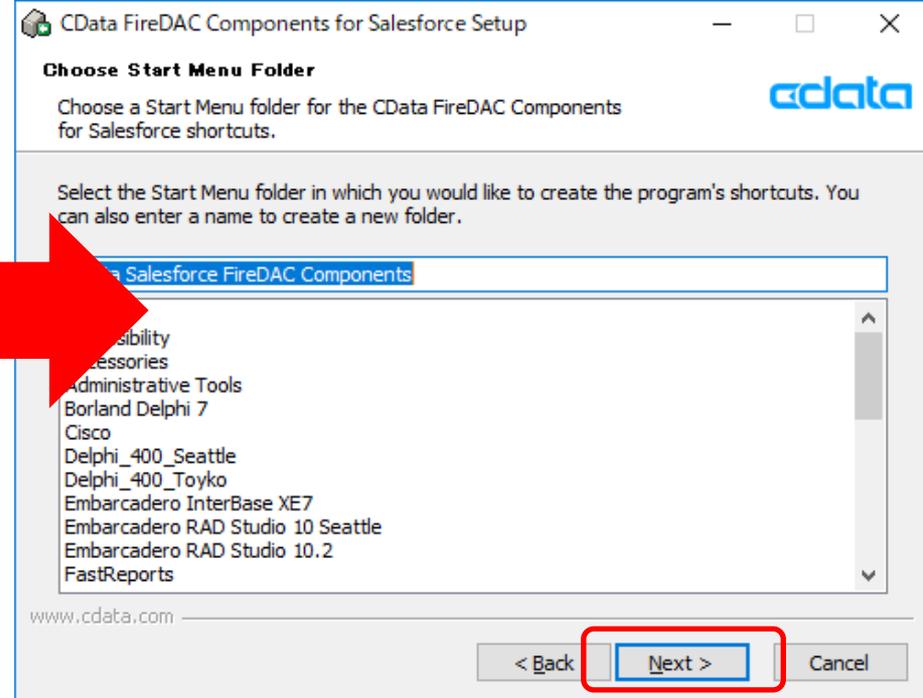
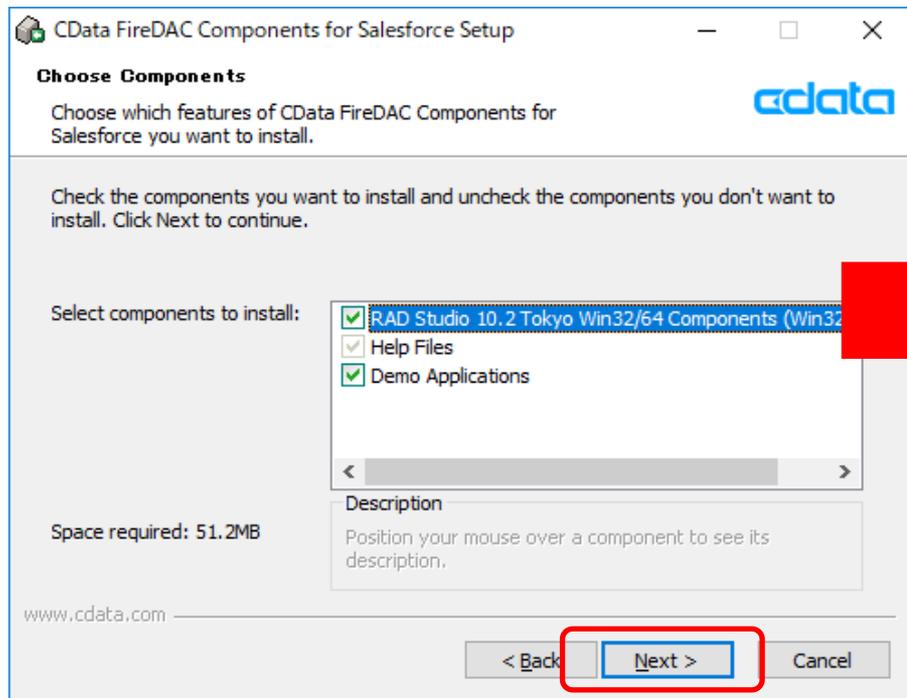
■ Enterprise Connectorsの組み込み例 (salesforce)

- 組み込み手順3
インストール先を設定



■ Enterprise Connectorsの組み込み例 (salesforce)

- 組み込み手順4
インストールオプションの選択



■ Enterprise Connectorsの組み込み例 (salesforce)

- 組み込み手順5
導入ユーザーの登録

CDat FireDAC Components for Salesforce Setup

Product Registration

Product registration is a requirement for support.

cd data

Name: Taisuke Yoshiwara *
Company: *
Title: *
Email: *****@****.**. **| *Phone Number: *
Address: *
City: * State: * Zip: *
Country: *

氏名とメールアドレスは必須

www.cd data.com

< Back Next > Cancel

CDat FireDAC Components for Salesforce Setup

Trial License Activation

cd data

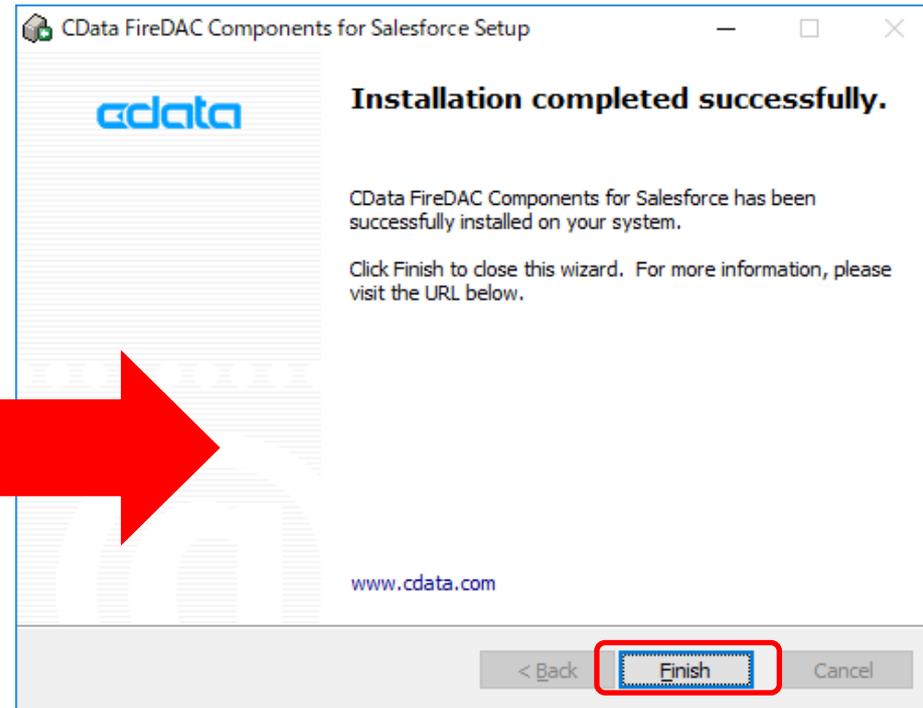
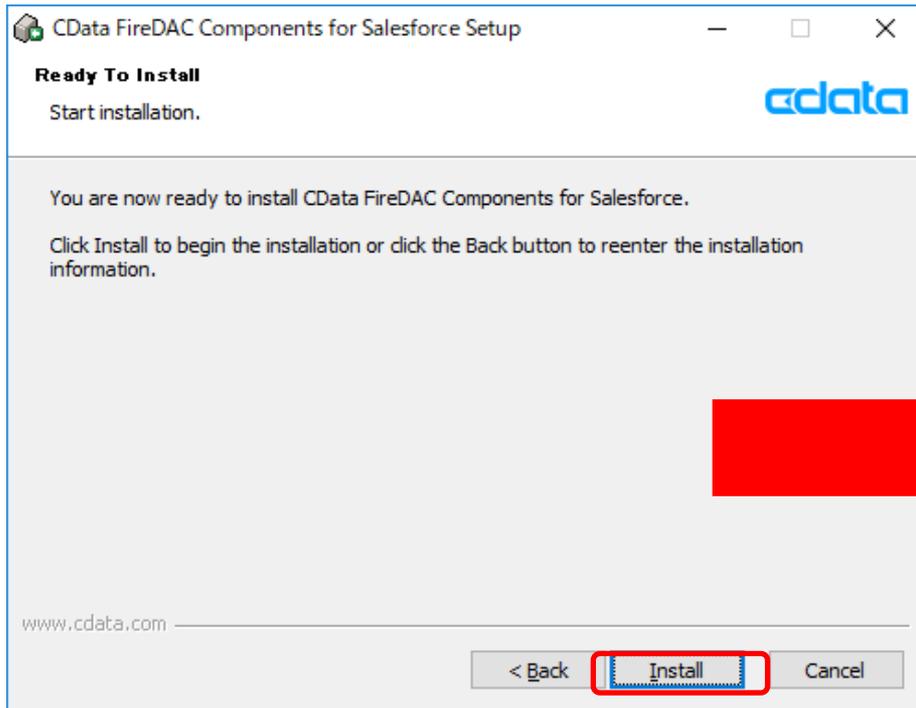
When you click 'Next', the setup will automatically activate a trial license.

www.cd data.com

< Back Next > Cancel

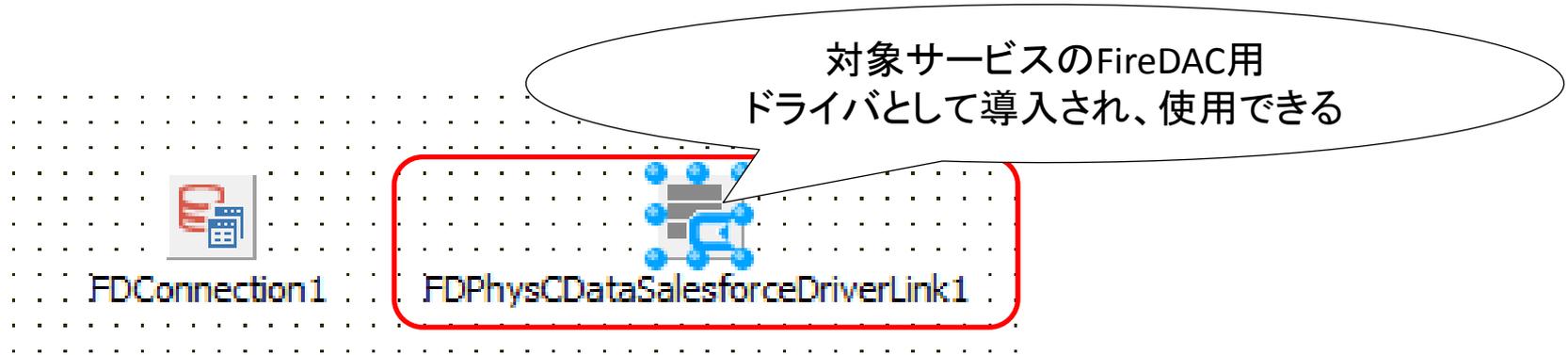
■ Enterprise Connectorsの組み込み例 (salesforce)

- 組み込み手順6
インストールの実行



■ Enterprise Connectorsの組み込み例 (salesforce)

• 組み込み手順7



定義 オプション 情報 SQL スクリプト

ドライバID(D):

接続定義名(N):

テスト(T) ウィ

パラメータ

DriverID

User

Password

通常DBと同じように
ドライバが選択できる

■ Enterprise Connectorsの組み込み例 (salesforce)

• 組み込み手順8

通常のFireDACのDB操作と同じ



定義 オプション 情報 SQL スクリプト

ドライバ ID(D): CDataSalesforce

接続定義名(N):

テスト(T)

パラメータ

パラメータ	値
DriverID	CDataSalesforce
User	*****@*****.**
Password	*****
SecurityToken	

例) ログインアカウントは
通常salesforceへログインする
アカウントを使用

TableList

- ContractHistory
- ContractStatus
- CorsWhitelistEntry
- CronJobDetail
- CronTrigger
- CspTrustedSite
- CustomBrand
- CustomBrandAsset
- Customers_Problem_c
- CustomObject2_c
- CustomObjectUserLicenseMetrics
- CustomPermission
- CustomPermissionDependency
- Dashboard
- DashboardComponent
- DashboardComponentFeed
- DashboardFeed
- DataAssessmentFieldMetric
- DataAssessmentMetric
- DataAssessmentValueMetric
- TableName: Salesforce.Account
- Tag: 0

SQLやテーブル名で、
通常のDBと同じように
アクセス・操作できる

Enterprise Connectorsの組み込み例 (salesforce)

実行

実際のsalesforceデータ

Delphi/400アプリでは自由にSalesforceへアクセス

ホーム Chatter ファイル 取引先 取引先責任者 ケース ソリューション レポート

新規取引先一覧

レポート生成状況: 完了

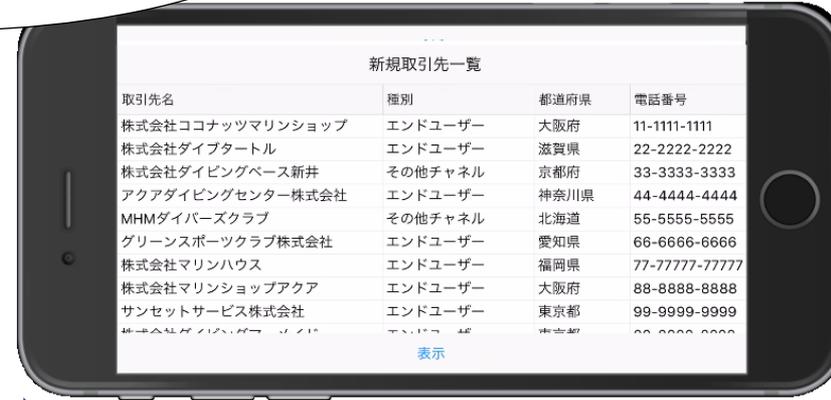
オプション:

集計情報: 表示 期間条件

レポート実行 詳細を非表示 カスタマイズ 保存 別名で保存 削除 印刷用に表示 詳細のエクスポート 登録

取引先所有者	取引先名	都道府県(請求先)	電話	種別	最終更新日
サポート	株式会社コナツツマリンショップ	大阪府	11-1111-1111	エンドユーザー	2018/05/12
サポート	株式会社ダイブスタート	滋賀県	22-2222-2222	エンドユーザー	2018/05/12
サポート	株式会社ダイビングベース新井	京都府	33-3333-3333	その他チャンネル	2018/05/12
サポート	アクアダイビングセンター株式会社	神奈川県	44-4444-4444	エンドユーザー	2018/05/12
サポート	MHMダイバーズクラブ	北海道	55-5555-5555	その他チャンネル	2018/05/12
サポート	グリーンスポーツクラブ株式会社	愛知県	66-6666-6666	エンドユーザー	2018/05/12
サポート	株式会社マリンハウス	福岡県	77-7777-7777	エンドユーザー	2018/05/12
サポート	株式会社マリンショップアクア	大阪府	88-8888-8888	エンドユーザー	2018/05/12
サポート	サンセットサービス株式会社	東京都	99-9999-9999	エンドユーザー	2018/05/12
サポート	株式会社ダイビングマーマイド	東京都	00-0000-0000	エンドユーザー	2018/05/12

総計 (10件)



IBM i

報告書の表示

顧客 NO	会社名
000001	1, 221 コナツツマリンショップ
000002	1, 513 ダイブハウスタートル
000003	3, 444
000004	
000005	
000006	

IBM i への同期や連携にも使える

■ Enterprise Connectorsを利用した拡張

• クラウドサービスやパッケージとの連携・同期例

クラウドサービスやパッケージ毎に
使い分けていたデータを同時に
利用できる。出力もアプリで自由

顧客マスターデータを
salesforce等のCRMを
中心に同期して一元化



kintone

会計パッケージ

salesforce

CRM

他DB

IBM i

中間サーバ

サービスやDBの導入や
接続設定は
中間サーバのみでOK

モバイルアプリでの利用も
Delphi/400ならマルチデバイ
ス開発で容易

クライアント



■ Enterprise Connectors利用時の補足

- 補足：通常のRDBMSと異なる点
 - ビューやプロシージャなど、用意されているものしか基本使えない。
(自由に作成はできない)
 - トランザクションは基本使えない。
 - ネットワークやサービス側の負荷が応答パフォーマンスに影響する。
 - 必要なデータだけを絞り込む(select * 的な検索は行わない)

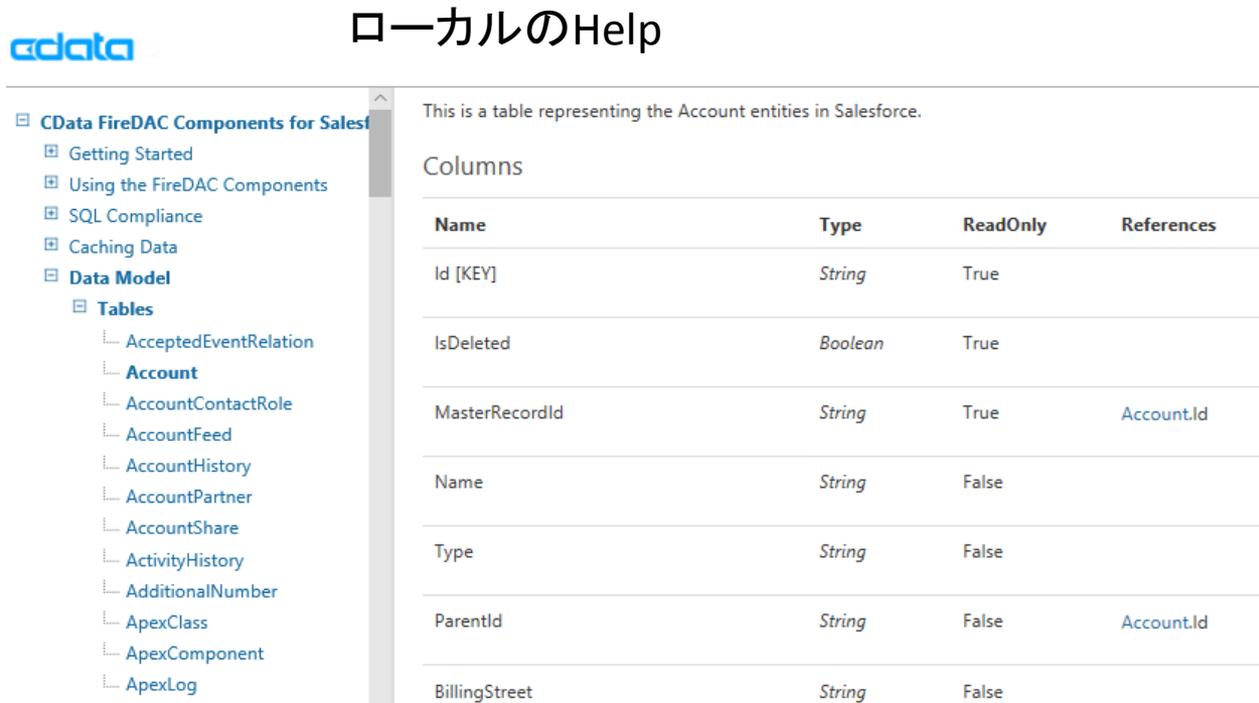
■ Enterprise Connectors利用時の補足

- 補足：テーブル等の定義

アクセスできるテーブル（の形で用意された）定義等はローカルのヘルプに詳しく掲載されている。

定義だけで分からない不明なものは実データを見て判断する方法も有効。

cddata ローカルのHelp



This is a table representing the Account entities in Salesforce.

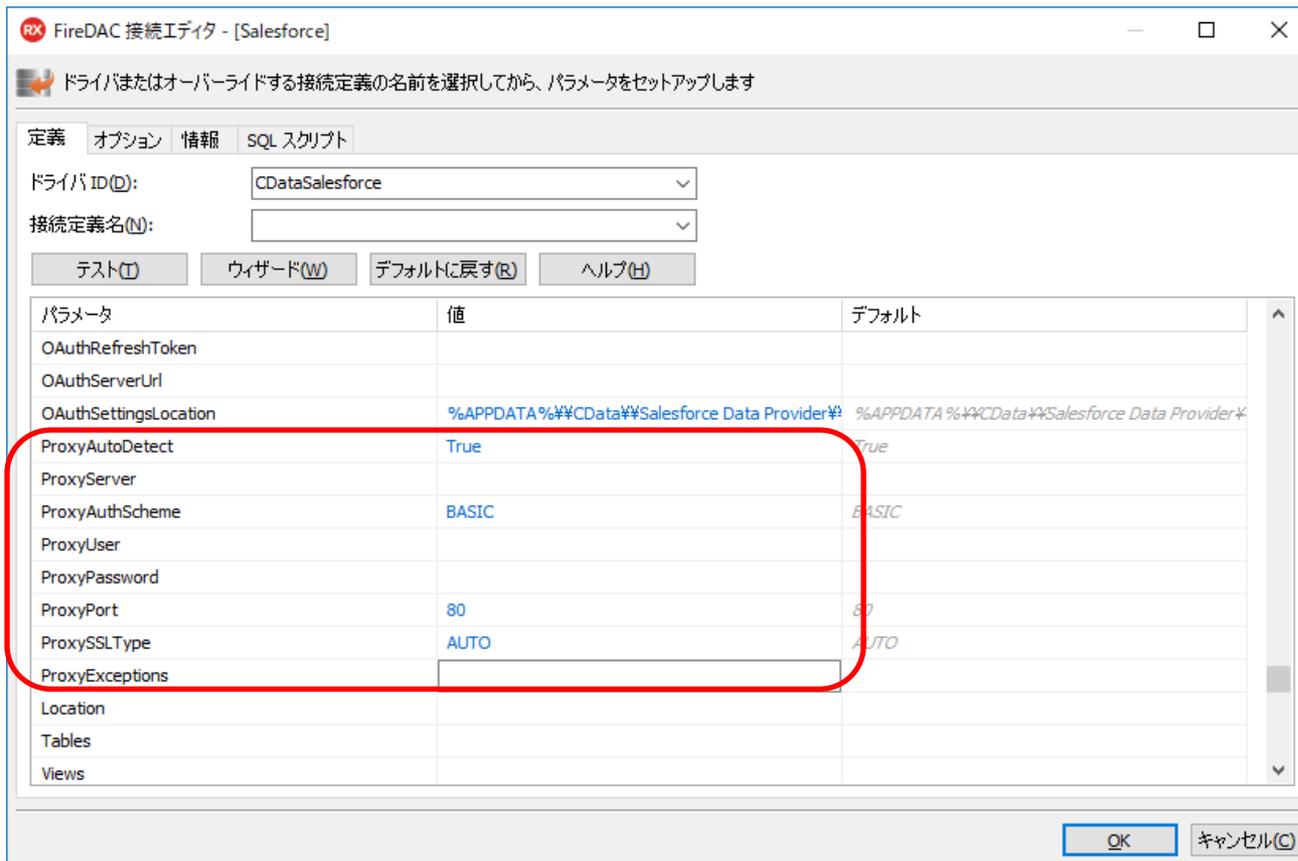
Columns

Name	Type	ReadOnly	References
Id [KEY]	String	True	
IsDeleted	Boolean	True	
MasterRecordId	String	True	Account.Id
Name	String	False	
Type	String	False	
ParentId	String	False	Account.Id
BillingStreet	String	False	

■ Enterprise Connectors利用時の補足

- 補足：プロキシ対応

プロキシの設定もTFDConnectionのパラメータで対応可能
(サーバを配置する環境によっては便利)



5. まとめ

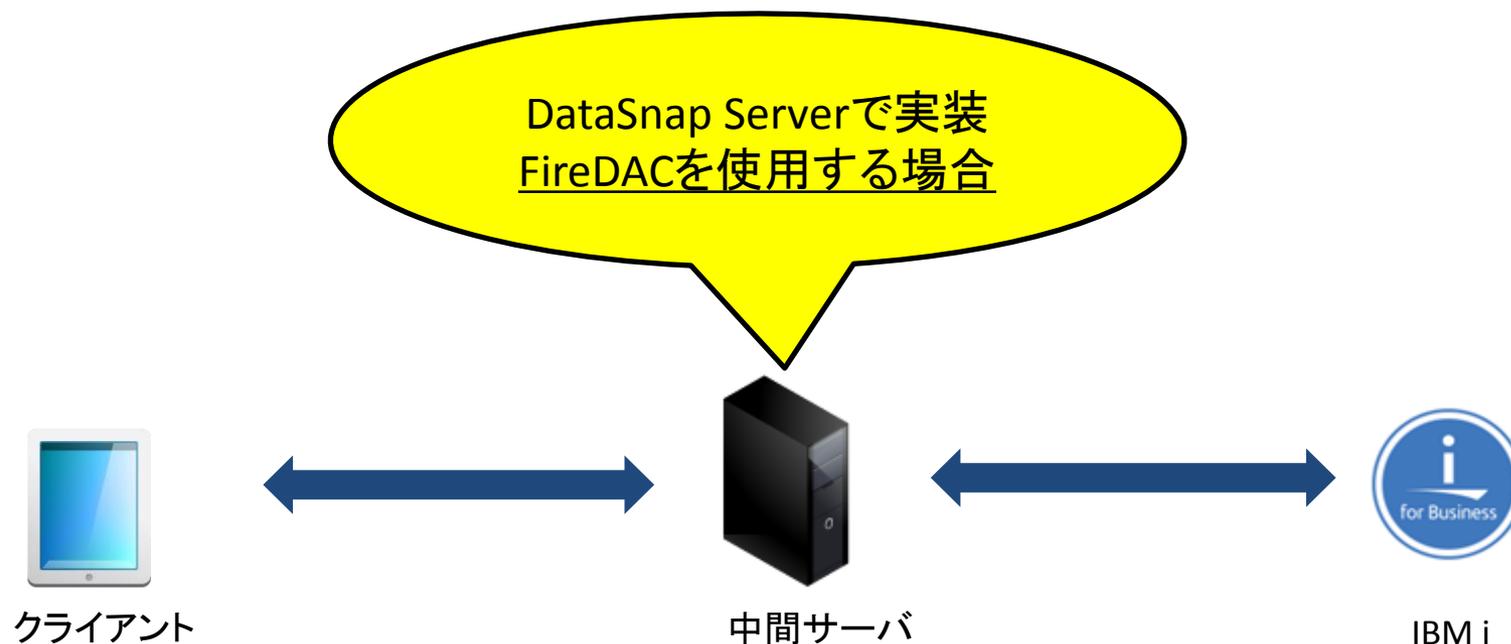
■ まとめ

- 多層アプリケーションによる機能の分離はメリットが多く、マルチデバイスでシステムを展開していく上では非常に有効。
- DataSnap Serverはサーバ開発用のSDK。開発できる自由度が高い。
- RAD Serverは標準で豊富な機能が提供されるサーバ。
- Enterprise Connectorsを中間サーバで利用すれば、クラウドサービスやパッケージなどの連携が容易にできる。

ご清聴ありがとうございました。

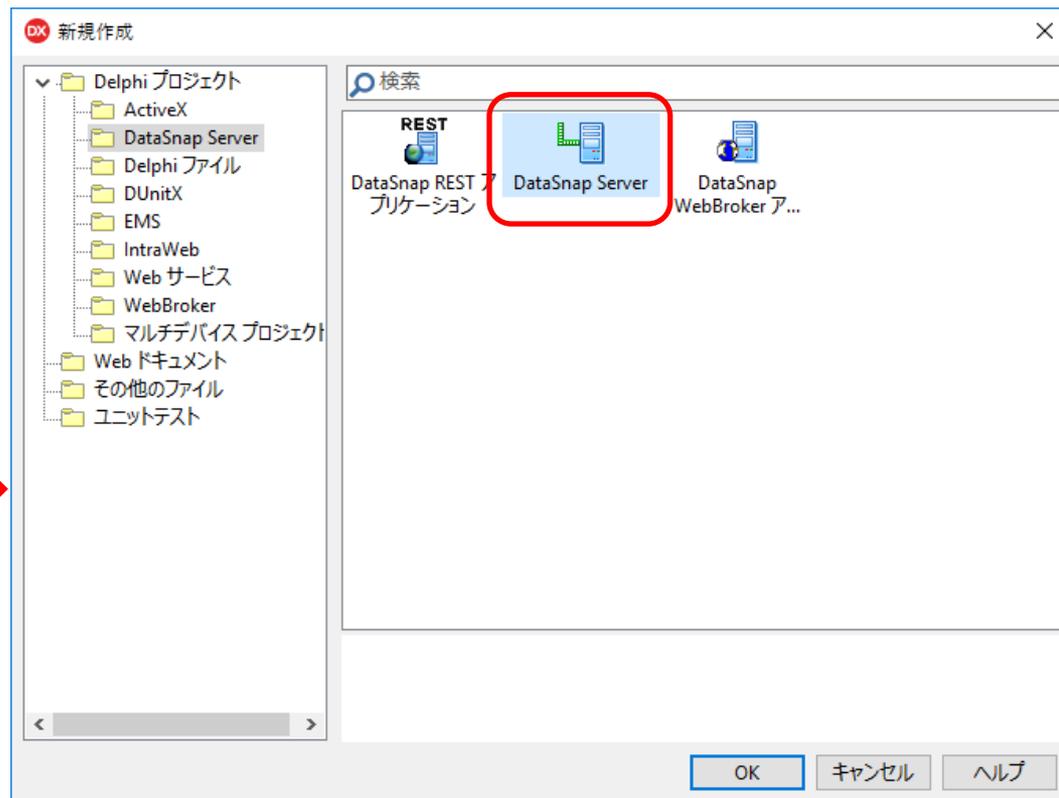
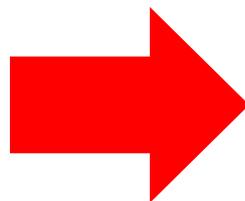
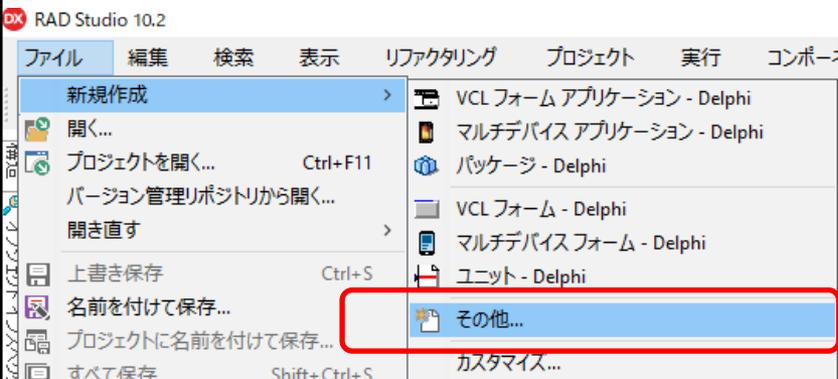
補足 : DataSnap Serverを使った実装手順 (FireDAC)

補足： DataSnap Serverを使った実装手順 (ウィザード)



補足： DataSnap Serverを使った実装手順（ウィザード）

- 実装手順1
プロジェクトの作成



補足：DataSnap Serverを使った実装手順（ウィザード）

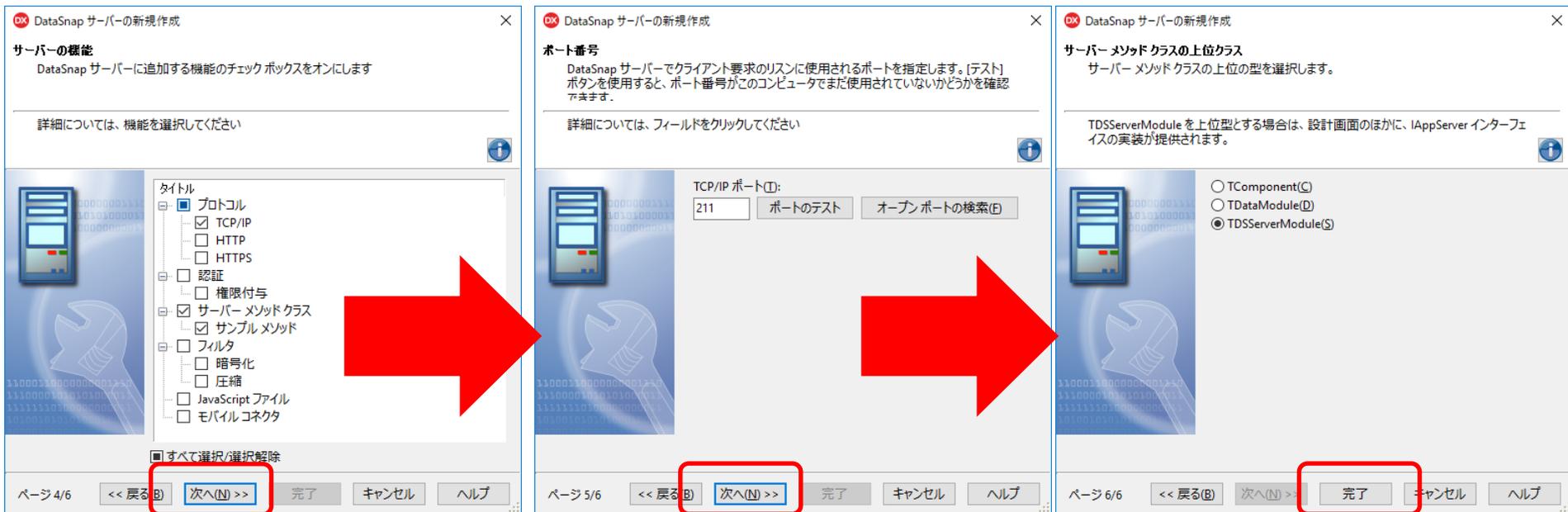
• 実装手順2 ウィザードの指定

テスト時はフォームアプリケーションが手軽だが、運用時はサービスアプリケーションが現実的（同じソースでプロジェクトだけ分けておくと便利）



補足：DataSnap Serverを使った実装手順（ウィザード）

• 実装手順3 ウィザードの指定



補足：DataSnap Serverを使った実装手順（ウィザード）

- 実装手順4
自動生成されるユニット

The screenshot shows the Delphi IDE interface. On the left is the Project Manager window titled "Project1.dproj - プロジェクトマネージャ". It displays a tree view of the project structure:

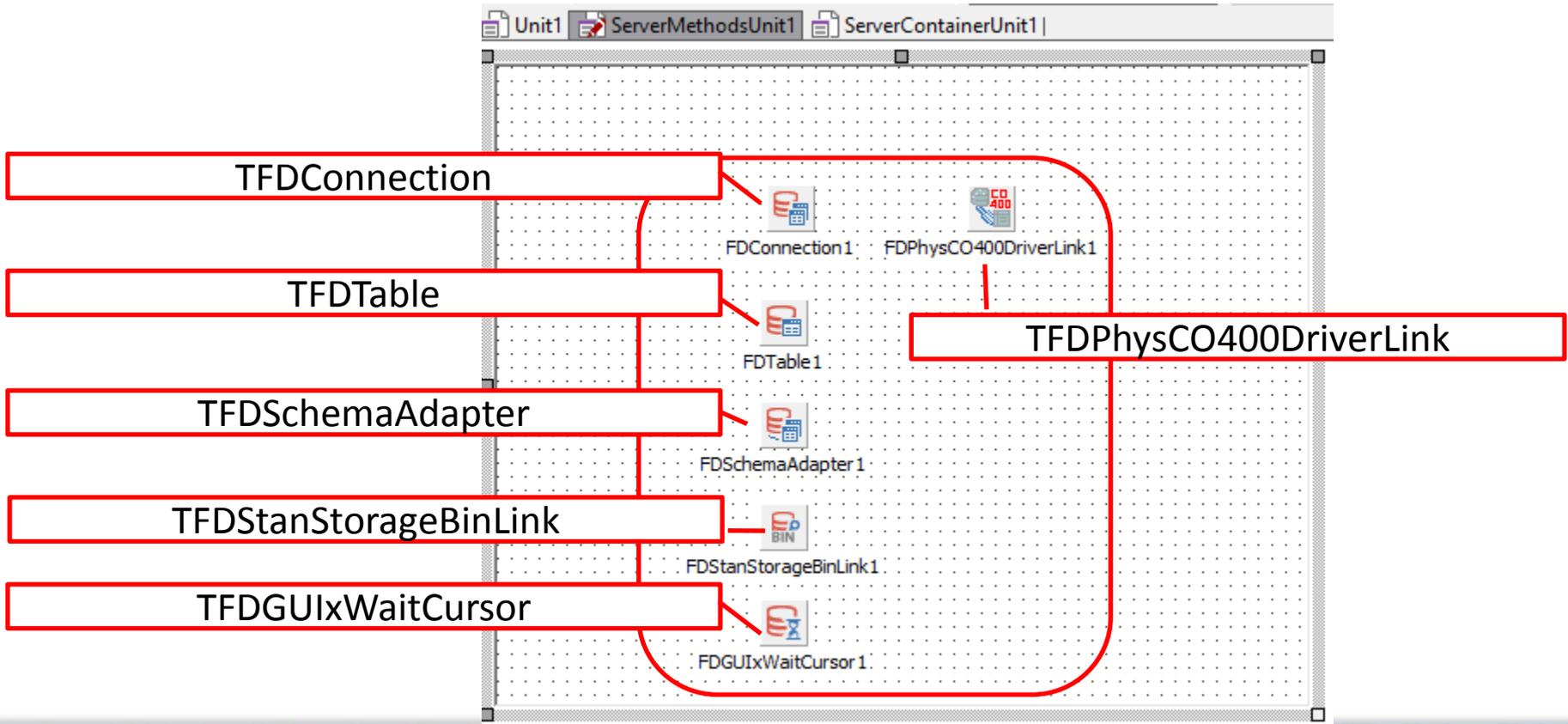
- ProjectGroup1
 - Project1.exe
 - ビルド構成 (Debug)
 - ターゲットプラットフォーム (Win32)
 - ServerContainerUnit1.pas
 - ServerMethodsUnit1.pas** (highlighted with a blue selection box)
 - Unit1.pas

On the right is the form editor window, showing a grid with a dotted pattern. The title bar of the form editor displays "Unit1 | ServerMethodsUnit1 | ServerContainerUnit1 |". A speech bubble points to the "ServerMethodsUnit1.pas" file in the Project Manager, containing the text:

Unit1はフォームアプリケーションを選択した場合に生成される。

補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順5
サーバ機能のコンポーネント配置
ServerMethodsユニットに機能を実装



補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順6
コンポーネントの設定
TFDConnection

FireDAC 接続エディタ - [FDConnection1]

ドライバまたはオーバーライドする接続定義の名前を選択してから、パラメータをセットアップします

定義 オプション 情報 SQL スクリプト

ドライバ ID(D): CO400

接続定義名(N):

テスト(T) ウォード(W) デフォルトに戻す(R) ヘルプ(H)

パラメータ	値	デフォルト
DriverID	CO400	CO400
Pooled	False	False
Database	POWER8	
User_Name	D4TEC	
Password	D4TEC	
MonitorBy		
ODBCAdvanced	LibraryOption=D4TEC22LIB	
LoginTimeout		
Alias		
Server		
Port		
ExtendedMetadata		
MetaDefSchema		
MetaCurSchema		

通常 of IBM i 接続設定

TFDTable

オブジェクト インスペクタ

FDTable1 TFDTable

検索

プロパティ イベント

Name	FDTable1
ObjectView	<input checked="" type="checkbox"/> True
ResourceOptions	(TFDBottomResourceOp...
SchemaAdapter	FDSchemaAdapter1
SchemaName	
TableName	CUSTOMER
Tag	0
Transaction	

フィールドエディタ... ビジュアルにバインド...
バインドソースの追加 クイック編集...

すべての項目が表示されています

補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順7
データ取得の機能を実装

usesにIPPeerClientを追加しておく。

```
function TServerMethods1.GetTable: TStream;  
begin  
  Result := TMemoryStream.Create;  
  try  
    FDTTable1.Close;  
    FDTTable1.Open;  
    //TFDSchemaAdapterを經由してStream形式で結果をセットする  
    FDSchemaAdapter1.SaveToStream(Result, TFDStorageFormat.sfBinary);  
    Result.Position := 0;  
  except  
    raise;  
  end;  
end;
```

補足 : DataSnap Serverを使った実装手順 (FireDAC)

DataSnap Serverで実装
FireDACを使用する場合



クライアント



中間サーバ

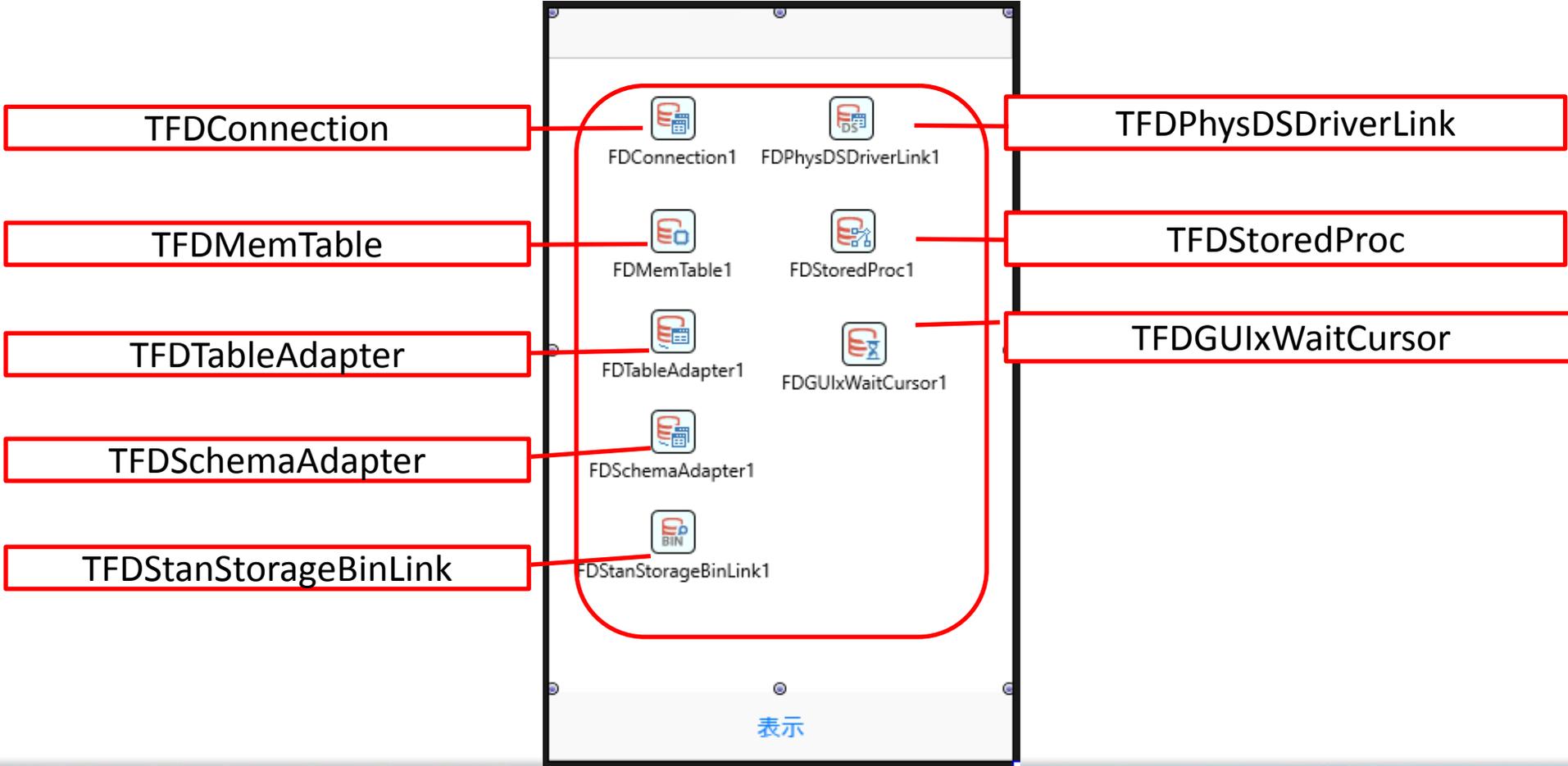


IBM i

補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順8

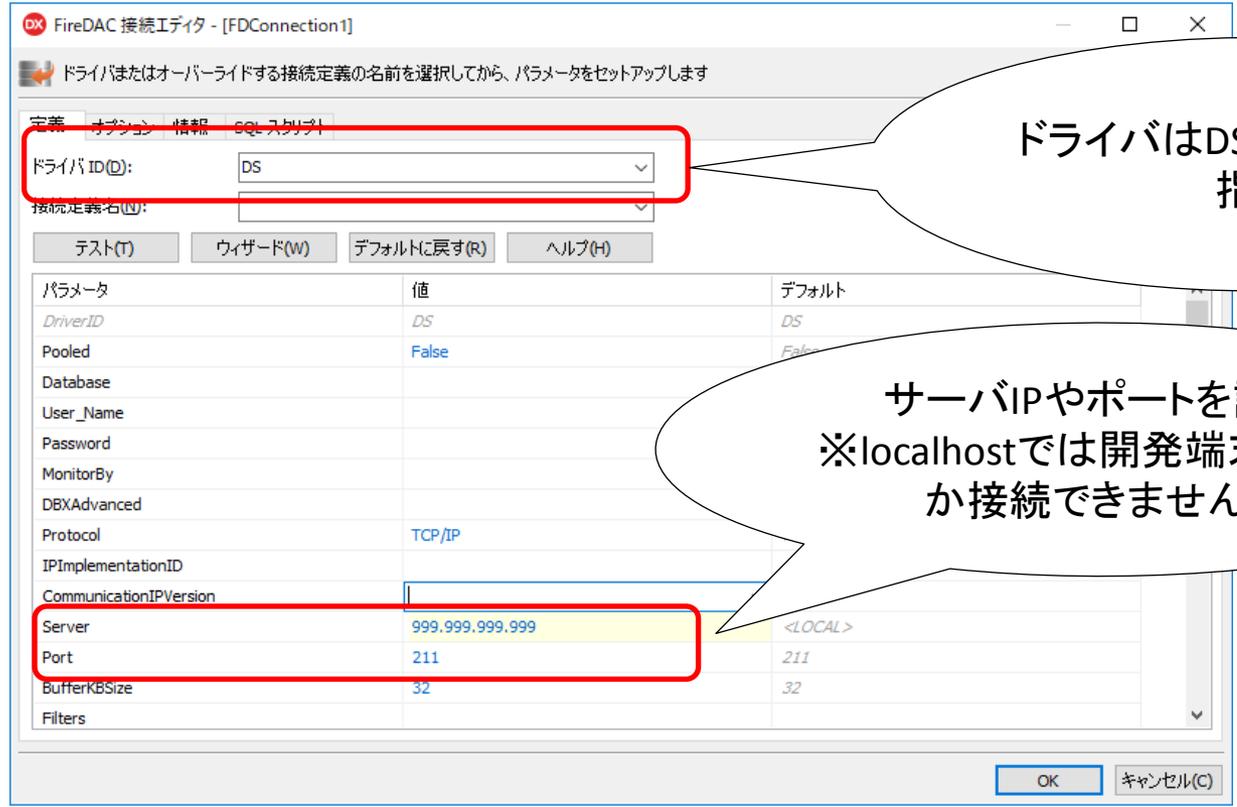
クライアントアプリのコンポーネント配置



補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順9
コンポーネントの設定

TFDConnection



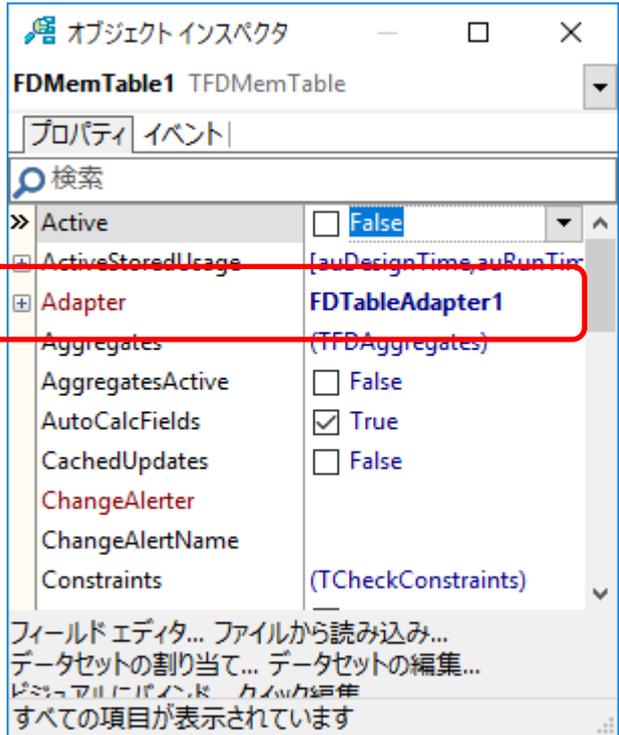
ドライバはDS(DataSnap)を指定

サーバIPやポートを設定
※localhostでは開発端末上
でしか接続できません。

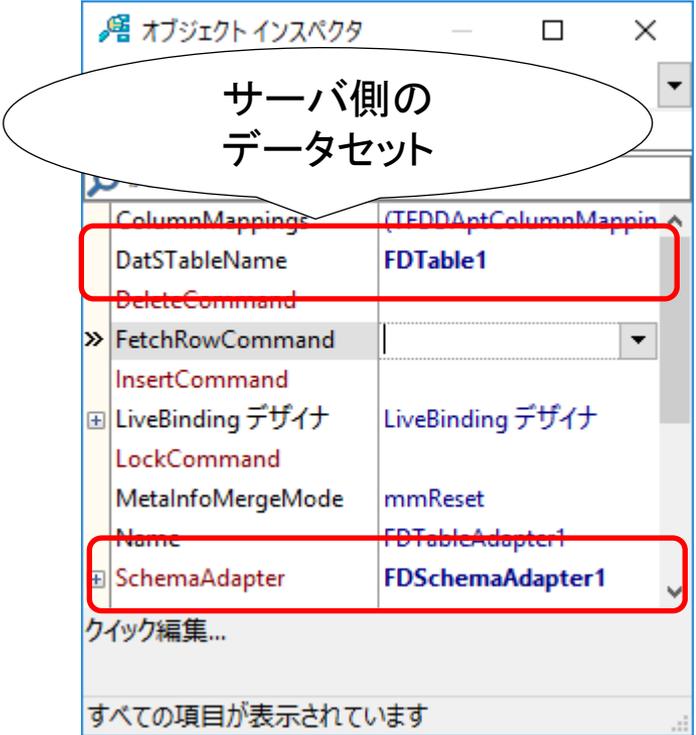
補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順10
コンポーネントの設定

TFDMemTable



TFDTableAdapter



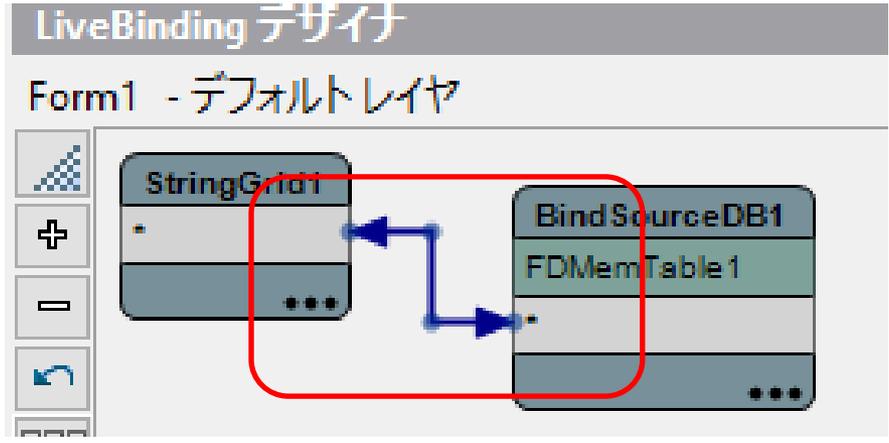
補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順11
コンポーネントの設定

TFDStoredProc



ライブバインディング設定



補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実装手順12

データ取得の機能を実装

```
procedure TForm1.Button1Click(Sender: TObject);
var
  LStringStream: TStringStream;
begin
  //TFDStoredProcでサーバで作成したfunctionをCallしてストリームを受け取る
  FDSStoredProc1.ExecProc;
  LStringStream := TStringStream.Create(FDSStoredProc1.Params[0].asBlob);
  try
    if LStringStream <> nil then
      begin
        LStringStream.Position := 0;
        FDSchemaAdapter1.LoadFromStream(LStringStream,
          TFDStorageFormat.sfBinary);
      end;
    finally
      LStringStream.Free;
    end;
  end;
end;
```

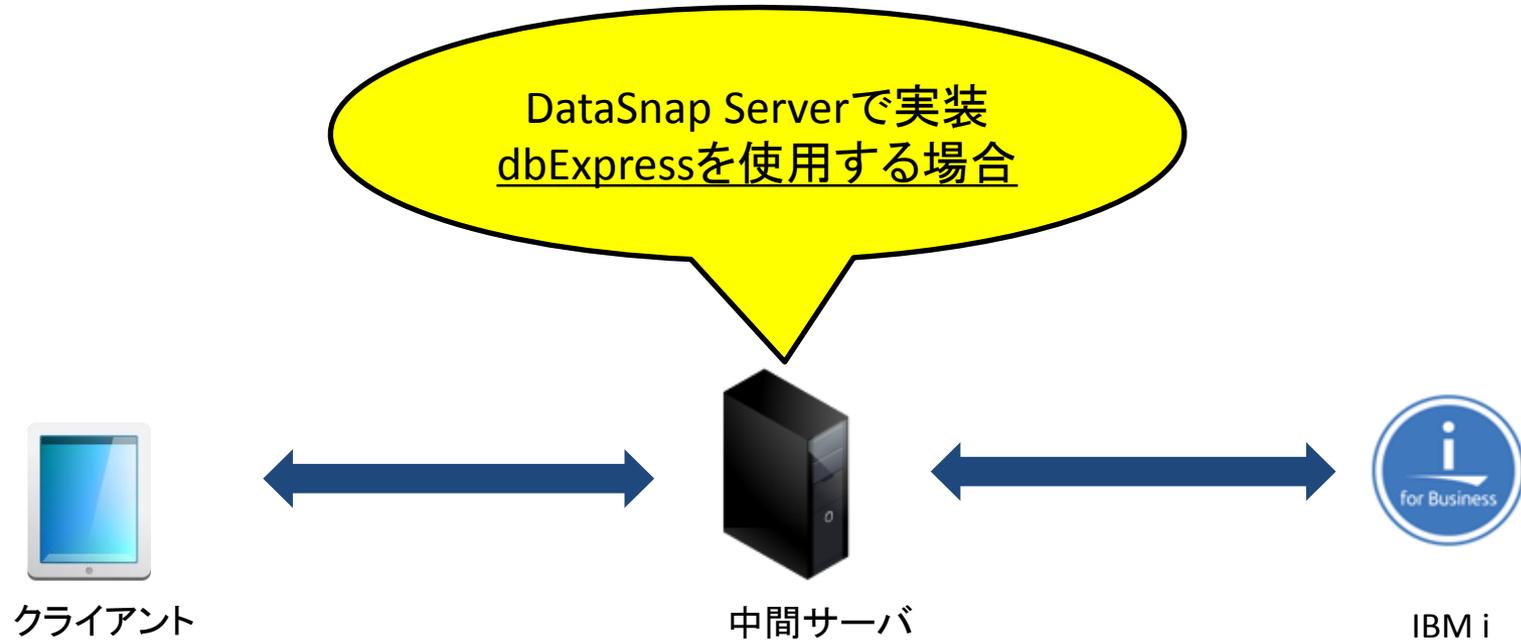
補足 : DataSnap Serverを使った実装手順 (FireDAC)

- 実行



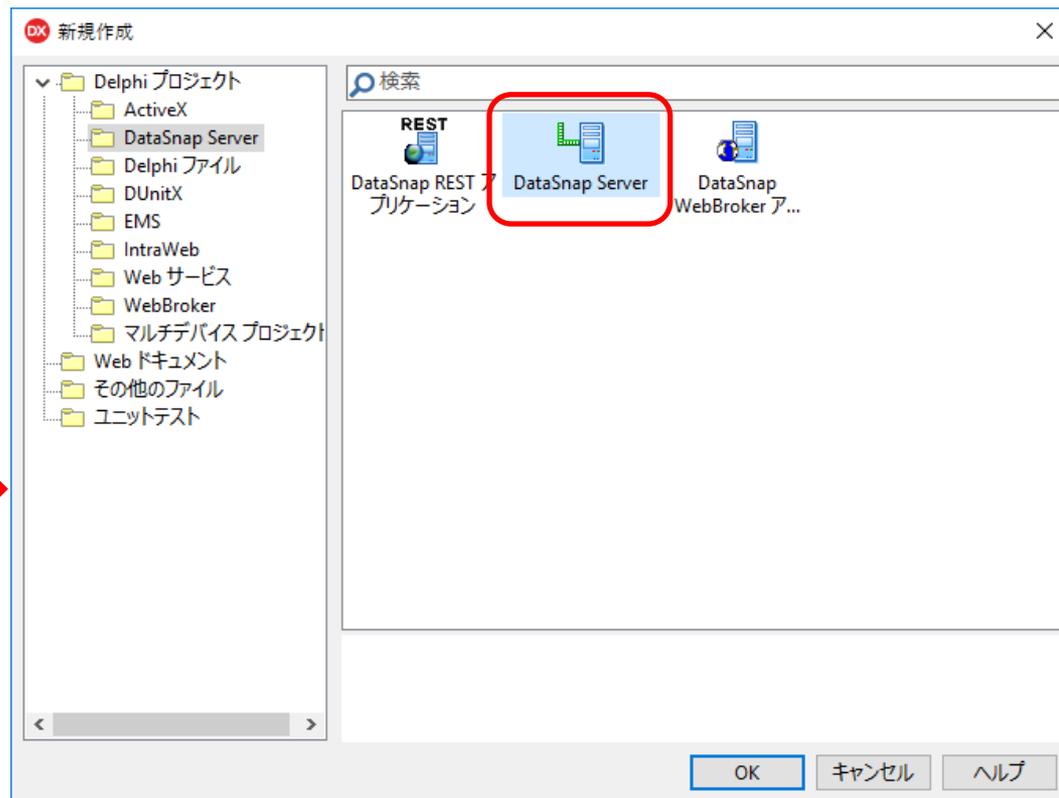
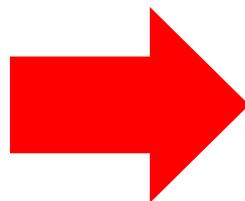
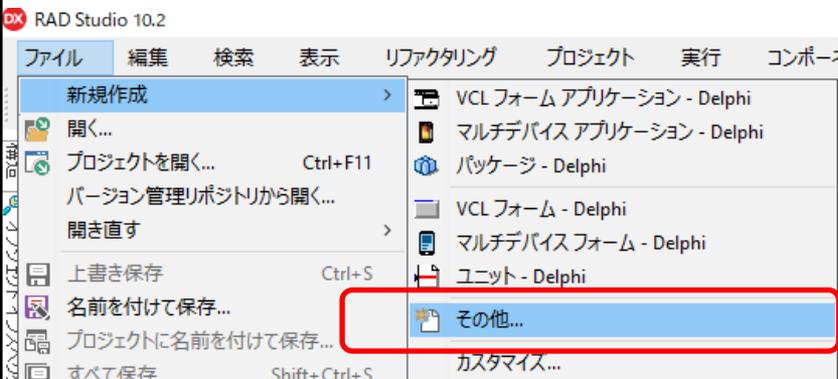
補足 : DataSnap Serverを使った実装手順 (dbExpress)

補足 : DataSnap Serverを使った実装手順 (dbExpress)



補足： DataSnap Serverを使った実装手順（ウィザード）

- 実装手順1
プロジェクトの作成



補足：DataSnap Serverを使った実装手順（ウィザード）

• 実装手順2 ウィザードの指定

テスト時はフォームアプリケーションが手軽だが、運用時はサービスアプリケーションが現実的（同じソースでプロジェクトだけ分けておくと便利）



補足：DataSnap Serverを使った実装手順（ウィザード）

• 実装手順3 ウィザードの指定



補足：DataSnap Serverを使った実装手順（ウィザード）

• 実装手順4

自動生成されるユニット

The screenshot shows the Delphi IDE interface. On the left is the Project Manager window titled "Project1.dproj - プロジェクトマネージャ". It displays a tree view of the project structure:

- ProjectGroup1
 - Project1.exe
 - ビルド構成 (Debug)
 - ターゲットプラットフォーム (Win32)
 - ServerContainerUnit1.pas
 - ServerMethodsUnit1.pas** (highlighted with a blue selection box)
 - Unit1.pas

On the right is the form editor window, which is currently empty. The title bar shows "Unit1 | ServerMethodsUnit1 | ServerContainerUnit1 |". A speech bubble points to the "ServerMethodsUnit1.pas" file in the Project Manager, containing the text:

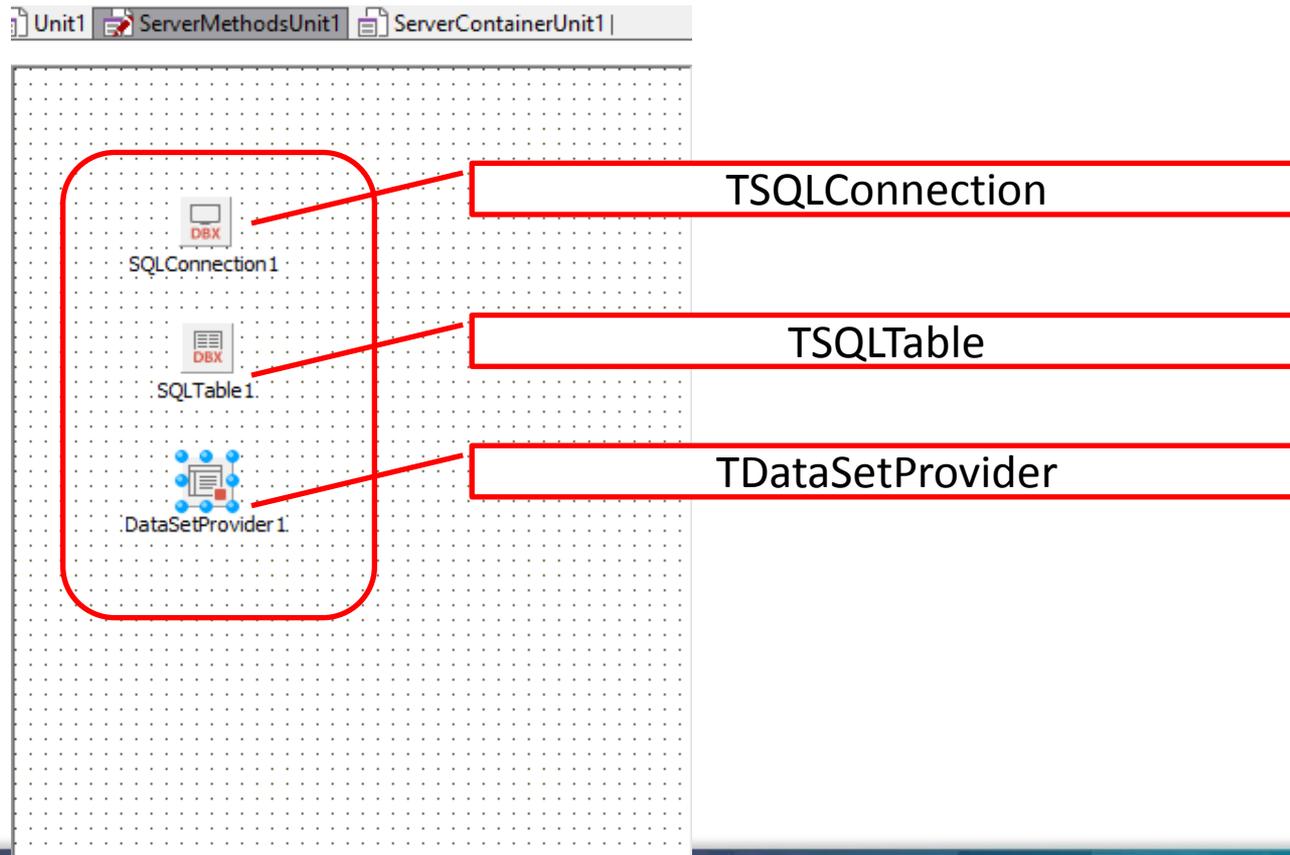
Unit1はフォームアプリケーションを選択した場合に生成される。

補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順5

サーバ機能のコンポーネント配置

ServerMethodsユニットに機能を実装



補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順6
コンポーネントの設定
TSQLConnection

FireDAC 接続エディタ - [FDConnection1]

ドライバまたはオーバーライドする接続定義の名前を選択してから、パラメータをセットアップします

定義 オプション 情報 SQLスクリプト

ドライバ ID(D): CO400

接続定義名(N):

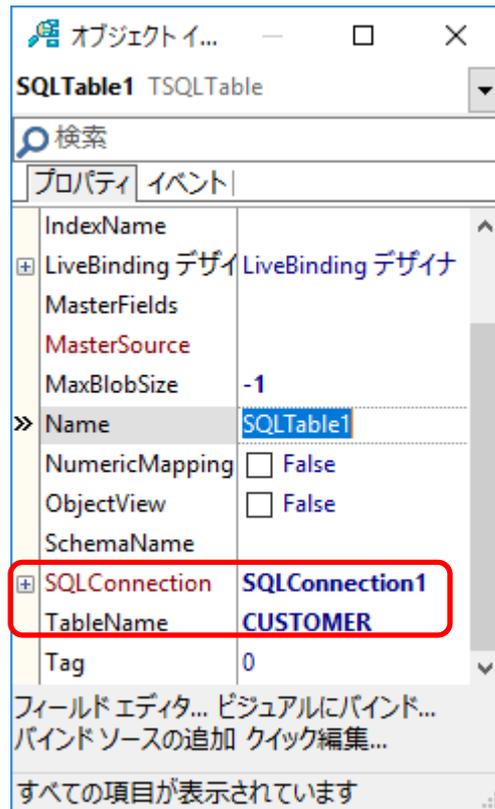
テスト(T) ウィザード(W) デフォルトに戻す(R) ヘルプ(H)

パラメータ	値	デフォルト
DriverID	CO400	CO400
Pooled	False	False
Database	POWER8	
User_Name	D4TEC	
Password	D4TEC	
MonitorBy		
ODBCAdvanced	LibraryOption=D4TEC22LIB	
LoginTimeout		
Alias		
Server		
Port		
ExtendedMetadata	False	
MetaDefSchema		
MetaCurSchema		

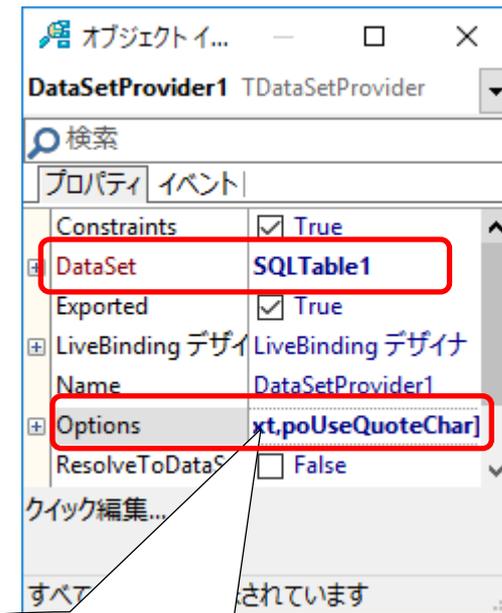
通常のIBM i 接続設定

補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順7
コンポーネントの設定
TSQLTable



TDataSetProvider



Optionプロパティの
"poAllowCommandText"を
"True"に設定しておくクライアント
側からSQLの指定も可能

補足 : DataSnap Serverを使った実装手順 (dbExpress)

DataSnap Serverで実装
dbExpressを使用する場合



クライアント



中間サーバ

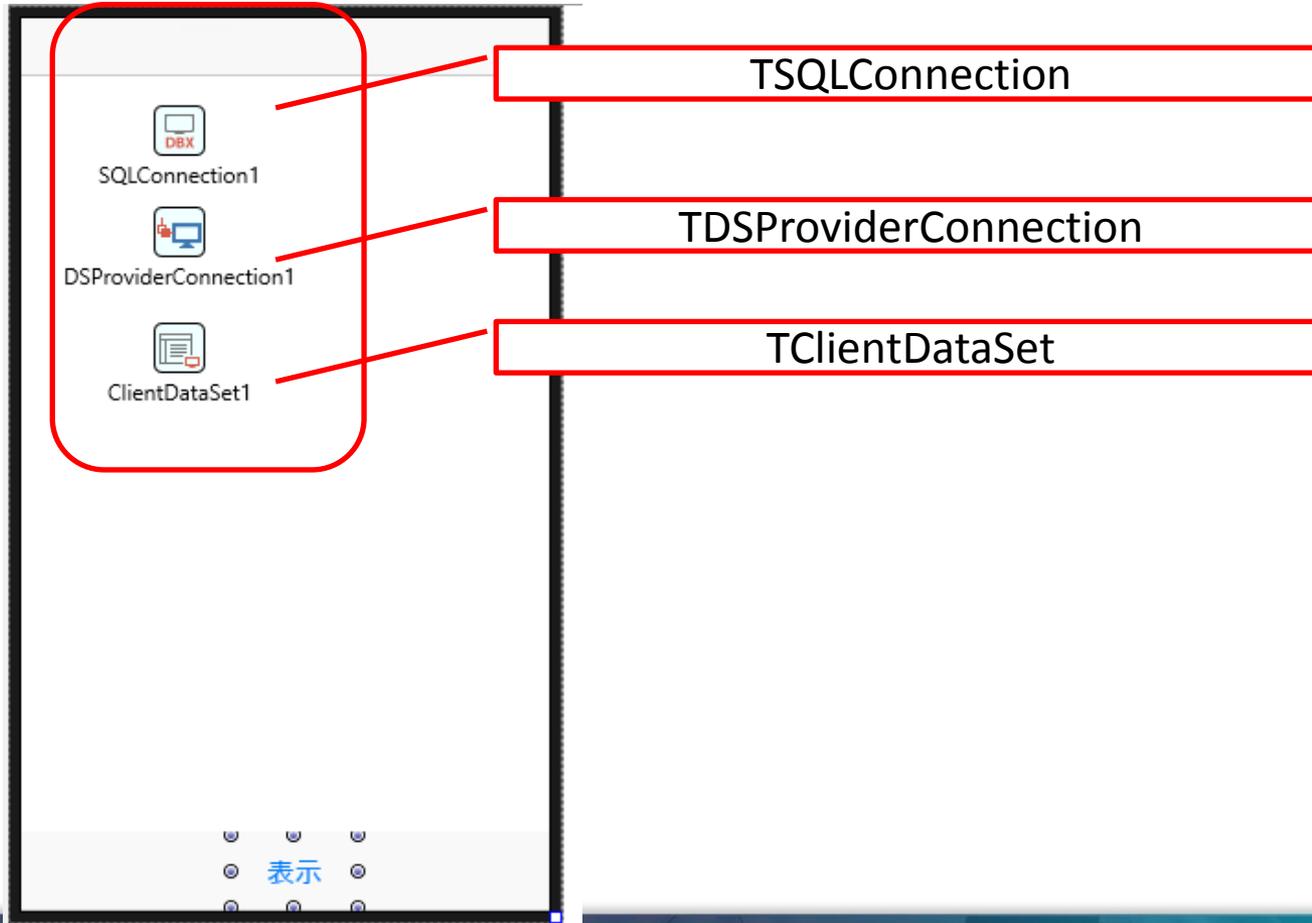


IBM i

補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順8

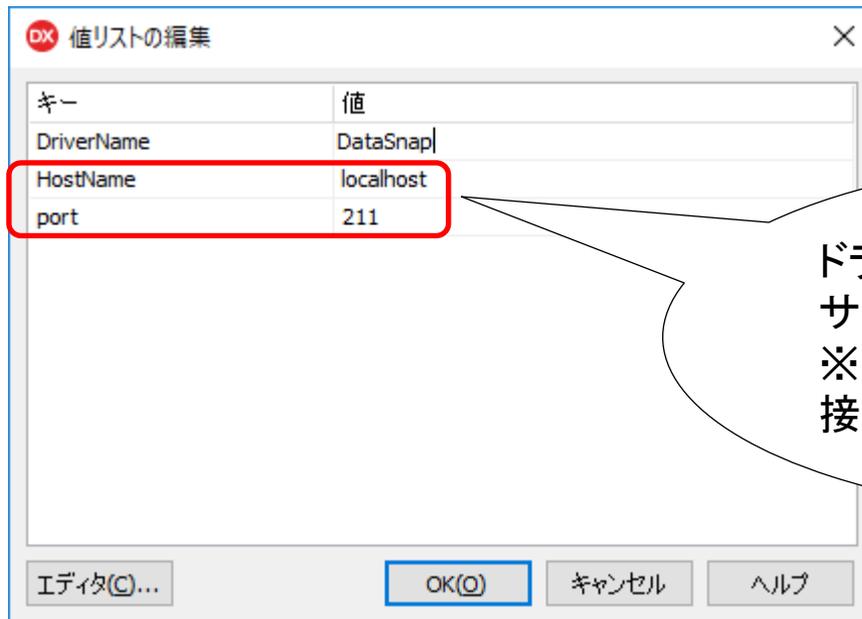
クライアントアプリケーションのコンポーネント配置



補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順9
コンポーネントの設定

TSQLConnection



ドライバはDataSnapを指定し、
サーバIPやポートを設定
※localhostでは開発端末上でしか
接続できません。

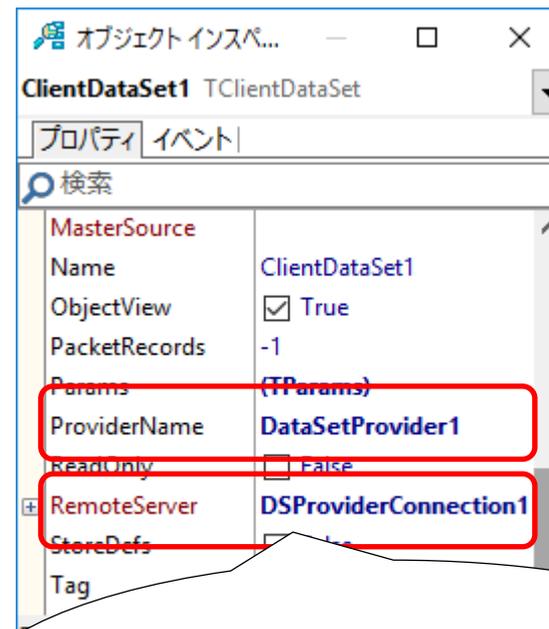
補足 : DataSnap Serverを使った実装手順 (dbExpress)

• 実装手順10

コンポーネントの設定
TDSProviderConnection



TClientDataSet



補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実装手順12

データ取得の機能を実装

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    //設定しているClientDataSetを開くだけ  
    ClientDataSet1.Open;  
end;
```

補足 : DataSnap Serverを使った実装手順 (dbExpress)

- 実行



補足 : DataSnap Serverを使った実装手順 (dbExpress)

- DBエンジンコンポーネントの組み合わせ

サーバ側をFireDACのコンポーネントを使い、クライアント側をdbExpressのコンポーネントで開発することも可能。

実装はコンポーネントが変わるだけでdbExpressの作り方とほぼ同じ。

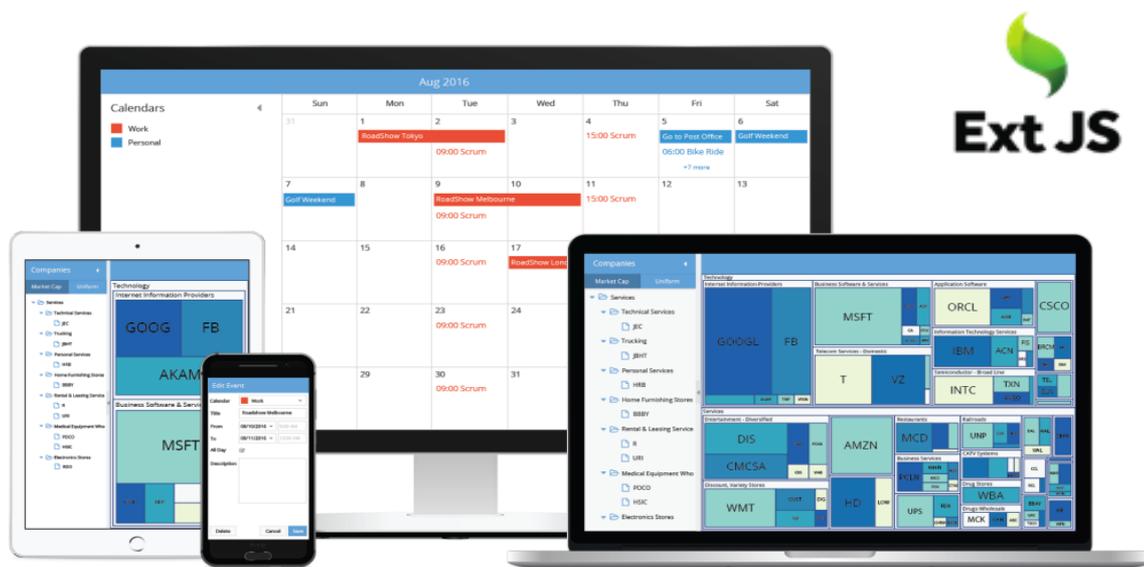
DataSnap Serverで可能な組み合わせ

組み合わせ	
クライアント側(サーバ接続)	サーバ側(DB接続)
FireDAC	FireDAC
dbExpress	FireDAC
dbExpress	dbExpress

補足：他言語アプリケーションへの連携

補足：他言語アプリケーションへの連携

- Delphi/400以外のアプリケーションでの利用
中間サーバから一般的なJSON形式でRESTの結果を提供すれば、
例えばSencha等で作成したWebアプリケーションで利用することも可能。



Sencha とは、Valence で使われる Web アプリケーションのビジュアル開発環境です。表、グラフ、ツリーなど、多彩なコンポーネントをドラッグ & ドロップすることにより、直観的に理解しやすい開発手法で設計・開発ができます。

補足：他言語アプリケーションへの連携

- 一般的なJSON形式でRESTの結果を戻す

```
procedure TCUSTResource1.Get(const AContext: TEndpointContext; const
ARequest: TEndpointRequest; const AResponse: TEndpointResponse);
var
  JSONResponse: TJSONObject;
  JSONArray: TJSONArray;
  JSONRecord: TJSONObject;
  aField: TField;
  count: Integer;
begin
  JSONResponse := TJSONObject.Create;
  JSONArray := TJSONArray.Create;
  count := 0;
```

補足：他言語アプリケーションへの連携

- 一般的なJSON形式でRESTの結果を戻す

```
// データセットの結果を1件ずつ取得してJSONArrayに追加する
FDTable1.Open;
while (not(FDTable1.Eof)) do
begin
    JSONRecord := TJSONObject.Create;

    for aField in FDTable1.Fields do
    begin
        JSONRecord.AddPair(LowerCase(aField.FieldName), aField.AsString);
    end;

    JSONArray.Add(JSONRecord);
    FDTable1.Next;
    inc(count);
end;

AResponse.Body.SetValue(JSONArray, True);
end;
```

補足：他言語アプリケーションへの連携

- 例：SenchaからDelphi/400と同様にデータを利用
Sencha Architect

ブラウザで実行

ID	会社名	都道府県	市区	住所1	住所2	郵便番号
1221	ココナツツマリンシ...	東京都	練馬区	大島町4-976-321	東京都	100-
1231	アクアダイビングセ...	北九州市	明太区曾根	541		808
1351	亀山ダイブセンター	千葉県	稲毛区	稲毛区亀山町632-1	稲毛区鶴亀2-4-22	263
1356	ダイビングハウスサ...	長崎県	佐世保市	松島町7-737	須佐町1163-1	857
1380	ダイブショップブル...	東京都	港区	鯉松町23-738	鯉3-15-23	105
1384	MHMダイバーズクラブ	千葉県	松戸市	埴輪町32		271
513	ダイブハウスタートル5	東京都	杉並区	東荻5-8-7		166
	ADVENTURE UNDERS...		GUAM	PO BOX 64594		9691
2118	グリーンスポーツ...	静岡県	清水市	中海老町633-21	東渡辺町3-147	424
2135	パイナップルダイバ...	沖縄県	石垣市	中村1455-1		907

Webアプリケーションでクロスドメインがネックになる場合は、RAD Server側のEMSServer.iniで設定するか、JSONPで返却するなどの工夫が必要