Web Data Service for Oracle ADF

Arkady Ganov

April 15, 2010 | Riga, Latvia

Speaker Qualifications

- Arkady Ganov, *IGK Service, Riga, Latvia*

  10+ years of appl. development with Oracle products
- Creator of Web Data Service for Oracle ADF
- ADF Swing with WDS applications since 2006
- Presentations:
  - "Web Data Service for Oracle ADF"

    IOUG Collaborate 2009, Orlando, FL, USA
  - "ADF Swing and Web Service Data Source"

    ODTUG Kaleidoscope 2007, Daytona Beach, FL, USA

IOUG independent oracle users group

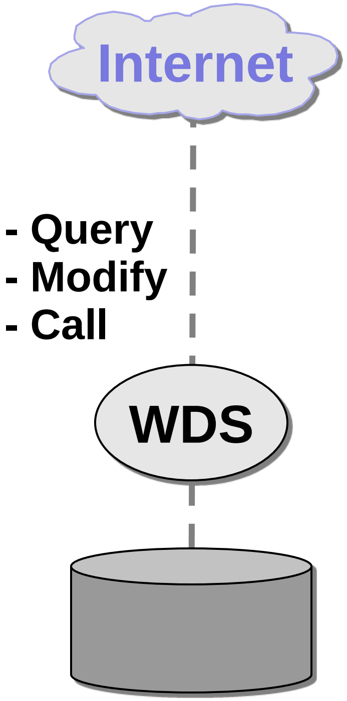
For the Complete Technology & Database Professional

Oracle Development Tools User Group

Session Objectives

- Show the way how to combine Oracle Application Development Framework with Web Data Service solution
- Show the benefits that application developers get from the ADF-WDS combination

What is Web Data Service?



- WDS = Web Service + Data Source
  - Exposes data source operations as services
- Makes a data source Internet enabled
  - Executes data source operations in response to incoming requests
- Single web service for all requests to the stored data and business logic

WDS's Role in Internet Applications

- Executing operations in remote data source
  - Internet access to the stored data and business logic
- New quality options from SOA/WS technology
  - abstraction and autonomy in data source operations
  - WS evolution to async calls, http session mngt, etc.
- Response to the new technological architectures (AJAX, MS Silverlight, Adobe Flash)
  - data is delivered at runtime separately from presentation
- Basis for Mashups, a new class of web apps gathering raw data from multiple sources

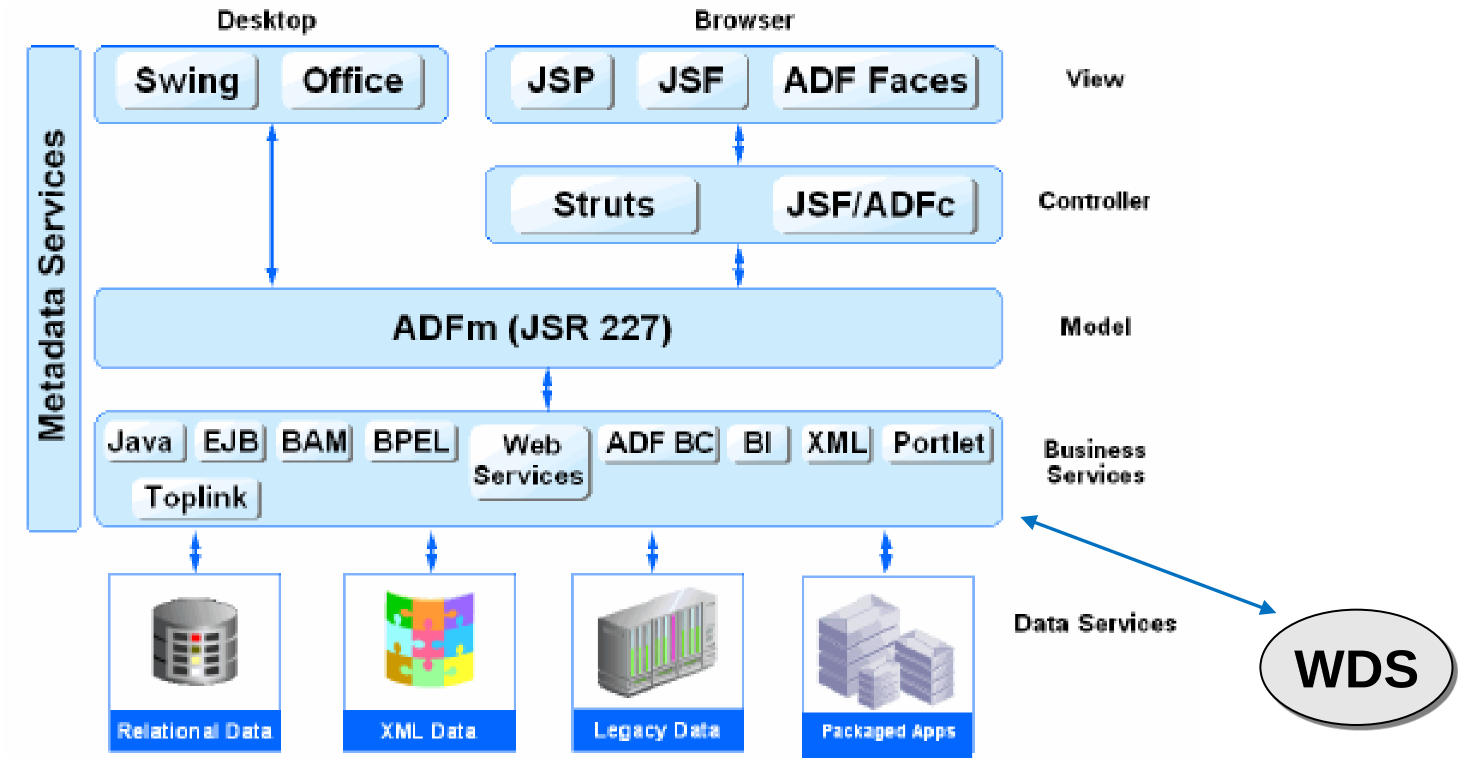
WDS Implementations

- Recent WDS examples:
  - DBWS from EclipseLink project (March 2009)
  - ADO.NET Data Services from Microsoft (mid 2008)
  - Data Web Services from IBM (late 2007)

- Commonly highlighted features:
  - Easy-to-develop, lightweight, yet robust solution
  - Storage independence (database, files, logic)
  - Abstract invocation of data source operations
  - SOAP and/or REST interface support

Can ADF Benefit from WDS ?

Figure 1–2 Simple ADF Architecture



- **WDS is a simple way for ADF applications to access remote data sources, regardless of client type**

Why Is WDS-for-ADF Possible ?

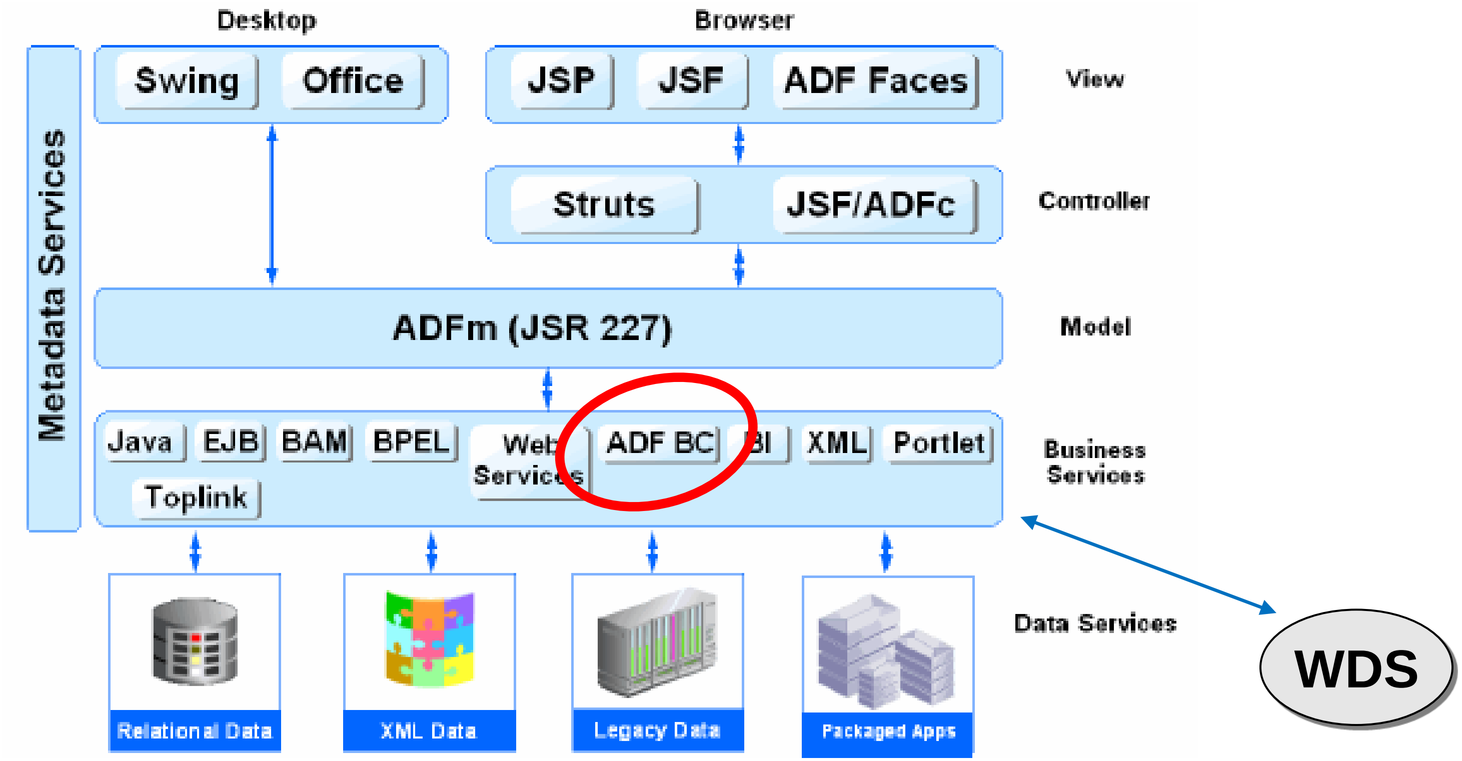
- ADF produces SQL statements for sending requests to database

☑ Get SQL from ADF ☑ Return results to ADF

- ADF has public methods for extracting generated SQL statements and associated parameters
- ADF provides well-documented technique for working with alternative data sources, i.e. how to return data into the framework components

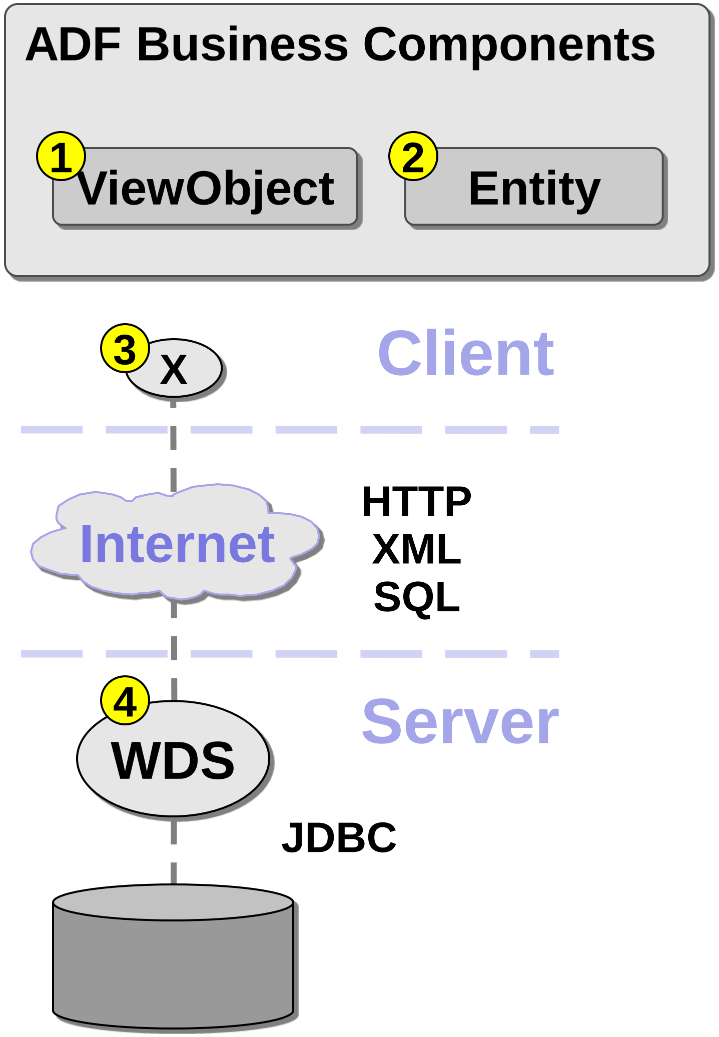
ADF Business Components Role

Figure 1–2 Simple ADF Architecture



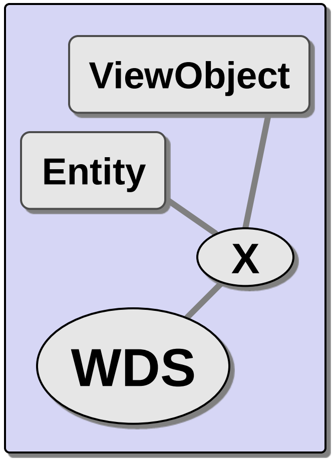
- **BC produce SQL statements for sending to database**

ADF Business Components and WDS



- ***ViewObject BC***: row collection (SELECT statements)
- ***Entity BC***: row changes (INSERT, UPDATE, DELETE)
- ***WDS Client***: transmits SQL and results between ADF and WDS
- ***WDS***: SQL requests processing in database

MyViewObject: Querying Data



- Fetches data collection from data source
- Extends oracle.jbo.server.ViewObjectImpl
- Overrides datasource methods following the technique from “ADF Developer’s Guide”
  - executeQueryForCollection():
    - Takes Query statement from ADF and passes to WDS Client for processing by Web Data Service
    - Receives resulting dataset from WDS Client and returns to ADF as new data collection

Getting Query SQL from ADF

```
SELECT deptno, dname, loc FROM dept
 WHERE loc = ? ORDER BY deptno;
```

executeQueryForCollection (…, Object[] *parameters*, …):

- getQuery()
- getWhereClause()
- getOrderByClause()
- *parameters*

String *statement*
Object[] *params*

- executeQuery

Returning Query Results to ADF

```
 DEPTNO    DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        30 SALES          NEW YORK
```

executeQueryForCollection(Object *qc*, …):

- setUserDataForCollection (*qc*, new Object[] {*data*, …})

- executeQuery

Object[][] *data*
int *queryHitCount*

WSDL: “executeQuery” Request

```
<xs:element name="executeQuery" >
 <xs:complexType>
  <xs:sequence>
   <xs:element name="statement“ type="xs:string" />
   <xs:element name="parameters“ type="tns:StringArray“/>
   <xs:element name="paramTypes" type="tns:StringArray“/>
   <xs:element name="maxFetchSize“ type="xs:int“ />
   <xs:element name="rangeStartIdx" type="xs:int” />
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

- java.sql.Types are used by WDS for converting parameters into their proper types and applying to the JDBC statement correctly

SOAP: “executeQuery” Request

```
SOAPAction: http://schemas.ag.lv/wds/executeQuery
...
<SOAP-ENV:Body>
  <executeQuery xmlns= ... >
    <statement>SELECT deptno, dname, loc FROM dept WHERE
loc = ? ORDER BY deptno</statement>
    <parameters>
      <string>NEW YORK</string>
    </parameters>
    <paramTypes>
      <string>12</string>
    </paramTypes>
    <maxFetchSize>-1</maxFetchSize>
    <rangeStartIdx>0</rangeStartIdx>
  </executeQuery>
</SOAP-ENV:Body>
```

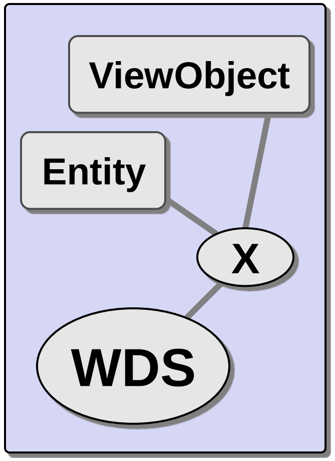
WSDL: “executeQuery” Response

```
<xs:element name="executeQueryResponse" >
 <xs:complexType>
  <xs:sequence>
   <xs:element name=“responseCode“ type=“tns:RespCodes" />
   <xs:element name=“executeResult“ type="xs:string" />
   <xs:element name=“columnNames“ type="tns:StringArray“/>
   <xs:element name=“columnTypes" type="tns:StringArray“/>
   <xs:element name=“searchResults“ >
    ...
      <xs:element name=“row“ type="tns:StringArray“
                                 maxOccurs="unbounded“ />
    ...
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

SOAP: “executeQuery” Response

```
<SOAP-ENV:Body>
  <executeQueryResponse xmlns="http://schemas.ag.lv/wds">
    <responseCode>OK</responseCode>
    <executeResult>2</executeResult>
    <columnNames> <s>DEPTNO</s> <s>DNAME</s> <s>LOC</s>
    </columnNames>
    <columnTypes> <s>2</s> <s>12</s> <s>12</s>
    </columnTypes>
    <searchResults>
<row> <s>10</s> <s>ACCOUNTING</s> <s>NEW YORK</s> </row>
<row> <s>30</s> <s>SALES</s> <s>NEW YORK</s> </row>
    </searchResults>
  </executeQueryResponse>
</SOAP-ENV:Body>
```

MyEntity: Modifying Data



- Posts row changes to data source
- Extends oracle.jbo.server.EntityImpl
- Takes from ADF:
  - entity post states {NEW, MODIFIED, DELETED}
  - attribute change indicators (isAttributeChanged)
- Datasource relevant methods:
  - postChanges(): explicitly call ADF to start DML operation
  - doDML(int *operation*, ...): override for building SQL and calling WDS Client

Getting UPDATE SQL from ADF

```
UPDATE dept SET dname = :1, loc = :2
 WHERE deptno = :3;
```

doDML(int operation, …):

- getEntityDef().getSource()
- isAttributeChanged(i) + getAttribute()
- getEntityDef(). getAttributeDefs()[i].getColumnName()
- getAttribute("Deptno")

WDS Client

String *statement*
Object[] *params*

- executeDML

WSDL: “executeDML” Request

```
<xs:element name="executeDML" >
 <xs:complexType>
  <xs:sequence>
   <xs:element name="statement“ type="xs:string" />
   <xs:element name="parameters“ type="tns:StringArray“/>
   <xs:element name="paramTypes" type="tns:StringArray“/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

- java.sql.Types are used by WDS for converting parameters into their proper types and applying to the JDBC statement correctly

SOAP: “executeDML” Request

```
SOAPAction: http://schemas.ag.lv/wds/executeDML
...
<SOAP-ENV:Body>
  <executeDML xmlns= ... >
    <statement>UPDATE dept SET dname = :1, loc = :2
WHERE deptno = :3</statement>
    <parameters>
      <s>MARKETING</s> <s>DALLAS</s> <s>30</s>
    </parameters>
    <paramTypes>
      <s>12</s> <s>12</s> <s>2</s>
    </paramTypes>
  </executeDML>
</SOAP-ENV:Body>
```

WSDL: “executeDML” Response

```
<xs:element name="executeDMLResponse" >
 <xs:complexType>
  <xs:sequence>
   <xs:element name=“responseCode“ type=“tns:RespCodes" />
   <xs:element name=“executeResult“ type="xs:string" />
  </xs:sequence>
 </xs:complexType>
</xs:element>


<xs:simpleType name=“RespCodes">
 <xs:restriction base="xs:string">
  <xs:enumeration value="OK" />
  <xs:enumeration value="NotAvailable" />
 </xs:restriction>
</xs:simpleType>
```

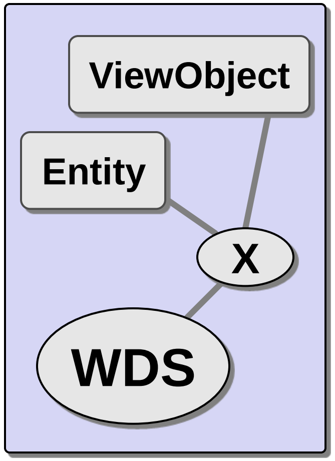
SOAP: “executeDML” Response

```
<SOAP-ENV:Body>
  <executeDMLResponse xmlns="http://schemas.ag.lv/wds">
    <responseCode>OK</responseCode>
    <executeResult>2</executeResult>
  </executeDMLResponse>
</SOAP-ENV:Body>
```

- <responseCode> clarifies non-failure executions {“OK”, “NotAvailable”, etc.}
- <executeResult> returns primary key of processed row in database

WDS Client: Bridge to WDS

- <u>Calls WDS on behalf of ADF BC requests</u>
- Converts data between ADF and SOAP
- Adds security tokens as SOAP header
- Uses HTTPS for message transportation

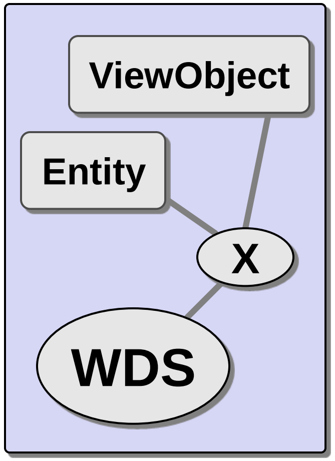


WSDL: User Authentication

```
<xs:element name=“authHeader" >
 <xs:complexType>
  <xs:sequence>
   <xs:element name=“username“ type="xs:string" />
   <xs:element name=“password“ type="xs:string" />
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

- Every SOAP request has user authentication header for WDS to get database connection
- SSL should be used in order to encrypt and protect transmitted messages

WDS: Port to Data Source



- Executes data source operations in response to incoming requests
- Invokes JDBC calls to data source
- Uses security tokens from SOAP header for opening connection to data source
- Provides HTTPS connection for client
- Completes every data source operation (*autonomy*)

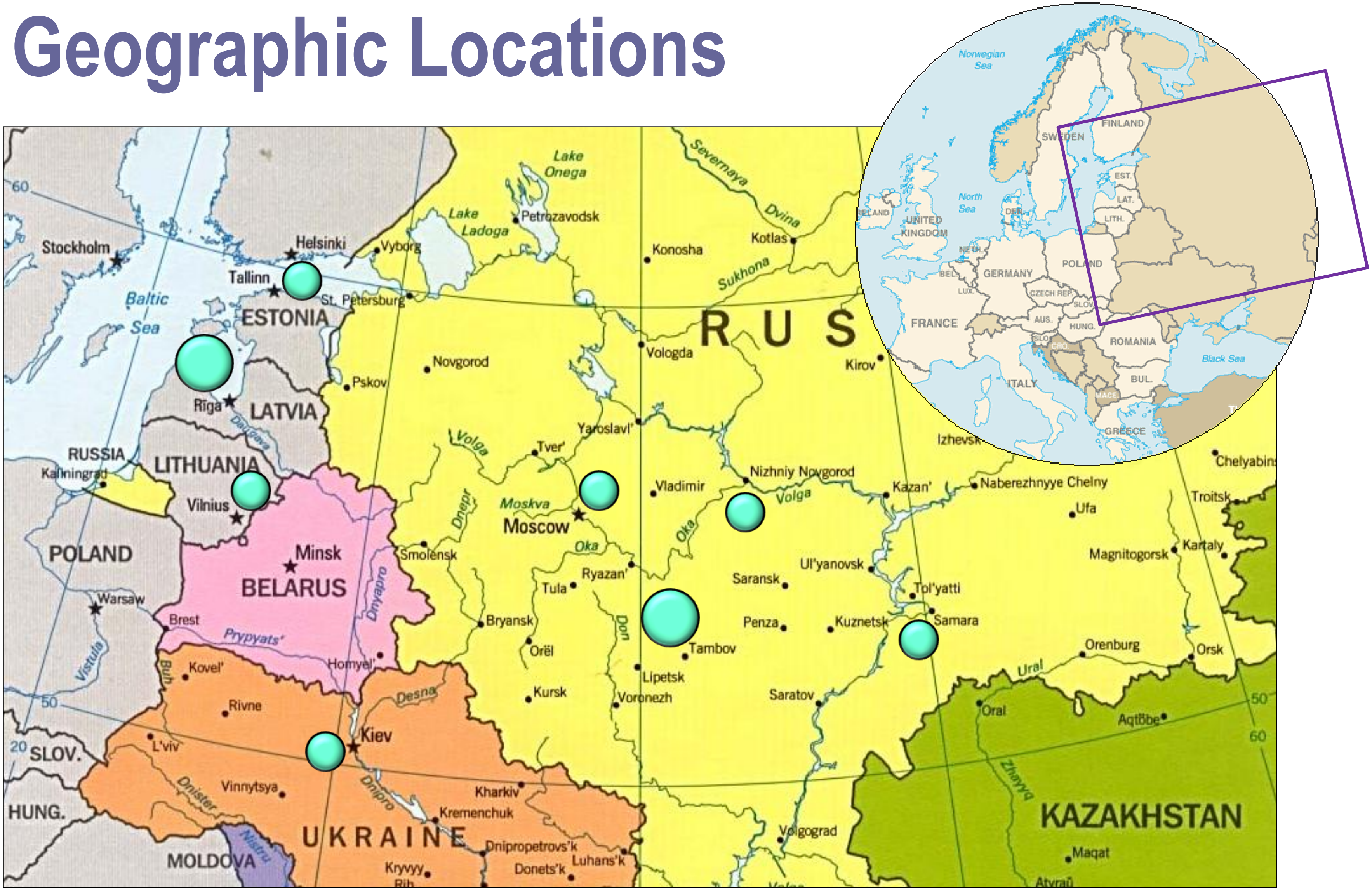
Conclusion

- **WDS provides ADF with simple and uniform way for accessing remote data sources**
- **WDS can be easily implemented and plugged into ADF in the low Business Components level**
- **ADF applications get Internet enabled regardless of client presentation and type of remote data source being consumed**
- **Single WS per data source is enough for all requests to the stored data and business logic**

Sharing Successful Experience

- Basis for data-centric Internet apps since 2006
- Suitable for intensive OLTP applications due to minimum network traffic during runtime
- 8 offices in 5 countries use the same versions, deployed by Java Web Start from central repository
- ADF Swing desktop and batch apps communicate with centralized database over HTTP
- Oracle products are enough for complete cycle from development to production

Geographic Locations



Thank You

- E-mail: arkady_ganov@igkservice.lv
- www.igk-group.com

igk®
group