

Data Access Patterns

Overview

- How does the Dataset impact common architectural goals?
- How do Datasets fit into a "SOA World" ?
- Enterprise Patterns for Data Access

Assumptions on Experience

- Intermediate to advanced ...
 - C# or VB.Net
 - ADO.Net and T-SQL
 - Datasets
 - Web Services
- Experience with Smart-Clients or ASP.Net

Products used in Demos

- Visual Studio 2005 beta 2
 - .Net Framework 2.0
- SQL Server 2005 beta

About Me

- Rob Daigneau
 - Senior Architect for Concentra
 - 15+ years in IT
 - **Enterprise-class database-driven applications**
 - Misc. Industries
 - **Manufacturing, Financial Services, Retail, Wholesale, Health-care**
 - **Internal apps, Commercial products**

Data Access Patterns

Common Application Layers

- Most business applications are primarily concerned with CRUD
 - **Writing to and reading from disk**
- Layers provide a means to organize this logic
 - Centralizes logic by purpose or subject matter
 - “1 stop shopping” for maintenance

Windows DNA

Presentation

Business Logic

Data Access

Data Management

A.K.A.

View

Domain Model

Data Access

Data Management

The Data Management Layer

Typical responsibilities include ...

- Hosts the **data model** for persistent storage
 - **Data entities (i.e. tables)**
 - **Data types (i.e. columns)**
 - **Data relations (i.e. referential integrity)**
 - **Data integrity (i.e. check constraints, triggers, etc.)**
- Manages writing to and reading from the data model
 - **Executes SQL and Stored Procs**
- Ultimately ensures the ACID principle is upheld

Data Model Design Decisions

Optimized for
Read
Performance

Optimized for Write
Performance and Data
Integrity



- Techniques
 - Normalization vs. Denormalization
 - Horizontal and Vertical Partitioning
 - Materialized Views
 - Etc.
- The data model is concerned with how best to store data

Observations About the Data Model

- Will probably change over time
- May serve many applications
- Implications
 - Excluding the most simple applications, we need to ...
 - **Have the flexibility to alter the data model**
 - **Minimize the impact changes to the data model have on other layers**
(i.e. reduce coupling between layers)
 - **Find code affected by data model changes with ease**

The Data Access Layer

- Typical responsibilities for this layer include ...
 - Knows identity of data sources and how to connect to data management layer
 - Knows how to perform CRUD operations
 - **Typically embedded SQL and Stored Procedures**
- May not be discrete layer
 - DAL logic oftentimes found in “Domain Objects”

The Domain Model Layer

- What is a “Business Object” (a.k.a. Domain Object)??
 - Represents an entity in the problem domain
 - **e.g. Customer, Order, Vendor**
- “Traditionally” has had these responsibilities ...
 - Manages temporary state
 - Invokes “Data Access” operations
 - Enforces data validation and executes “business rules”
 - Defines relationships between entities in the domain

Designing the Domain Model

In all but the simplest applications ...

- **Data Model != Domain Model**

- The data model defines how data is stored

- **Design is optimized to support fast reads or writes and data integrity**

- The domain model defines how data is used

- **Inheritance and Polymorphism play powerful roles**

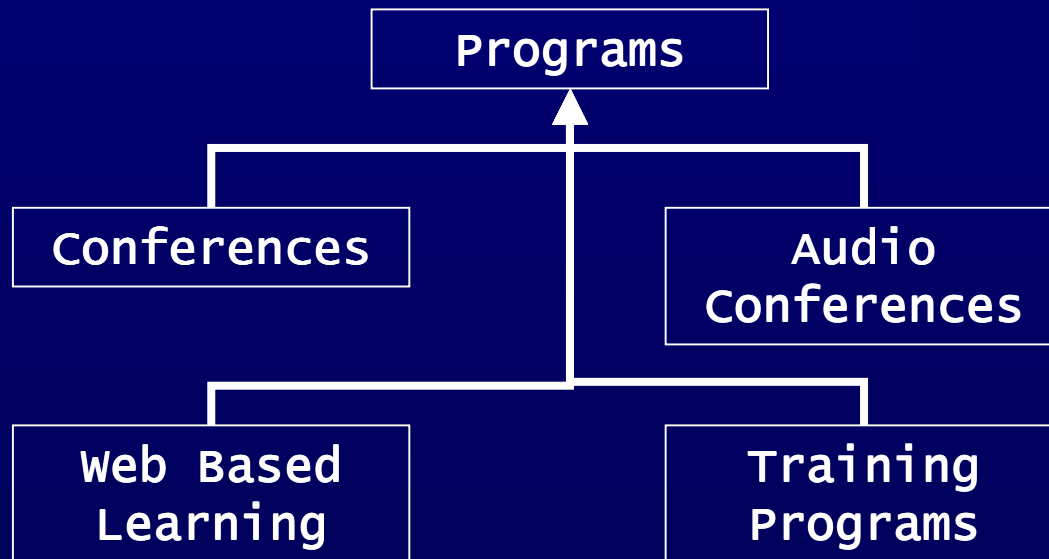
- **Loose coupling between the models promotes independent evolution**

- May not want 1:1 relationship between tables and objects

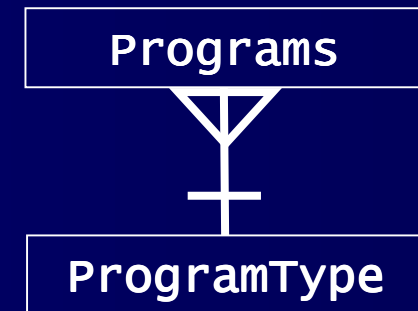
Data Model != Domain Model

This graphic illustrates how the domain model and data model can vary

Domain Model



Data Model



The Dataset

- What layers does the Dataset fit into?
 - Data Access Layer
 - Business Logic Layer
 - Replicates some Data Management Layer Features
- Very Table-Oriented
 - Structure typically based on the Data Model (i.e. the way data is stored)
- Business Objects (i.e. Domain Objects) provide an Object-Oriented alternative
 - Typed Datasets are an attempt to make Datasets more like “First Class Objects” in the Domain Model
 - Table Adapters improve upon the idea of Typed Datasets

Recap of Key Points

A few common goals for application architecture ...

- Ease of maintenance
 - Facilitated through layers
- Flexibility to change the data model
 - Achieved through loose coupling
- Ability to easily find and change code affected by data model changes
- Data Model \neq Domain Model

Beware the Ties that Bind

Or “Wizards can be Wicked”

Wizards

No-code approaches facilitate rapid development, but there are trade-offs ...

- Code redundancy
- Greater maintenance difficulties when data model changes
- Harder to control concurrency and transactions
- Decentralization of business rules
- Less flexible

ASP.Net 2.0 Wizards

```
selectCommand="SELECT * FROM [Customers]"></asp:SqlDataSource>
<asp:GridView ID="GridView2" runat="server" AutoGenerateColumns='
DataSourceID="SqlDataSource2" EmptyDataText="There are no dat
<Columns>
    <asp:BoundField DataField="ShipperID" HeaderText="Shipper
    <asp:BoundField DataField="CompanyName" HeaderText="Compa
    <asp:BoundField DataField="Phone" HeaderText="Phone" Sort
</Columns>
</asp:GridView>
<asp:SqlDataSource ID="SqlDataSource2" runat="server" ConnectionS
DeleteCommand="DELETE FROM [Shippers] WHERE [ShipperID] = @or
InsertCommand="INSERT INTO [Shippers] ([CompanyName], [Phone]
ProviderName=" <%$ ConnectionStrings:NorthwindConnectionString
SelectCommand="SELECT [ShipperID], [CompanyName], [Phone] FRE
<InsertParameters>
    <asp:Parameter Name="CompanyName" Type="String" />
    <asp:Parameter Name="Phone" Type="String" />
</InsertParameters>
<UpdateParameters>
```

- Like stepping back to the "Classic ASP" spaghetti-code days
 - A Mix of HTML and Data Access Logic
 - If data model changes, all embedded SQL must be found and altered
- Violates MVC Pattern

The Windows Forms Data Sources Wizards

- The “Data Sources” window is a really slick tool **BUT**
- The SQL gets buried in files that you can’t directly edit
 - Can only use a wizard
 - Think of how hard it would be to find and update code as the model changes

```
Me.m_adapter.DeleteCommand.Connection = Me.Connection
Me.m_adapter.DeleteCommand.CommandText = "DELETE FROM [dbo].[C
    "CompanyName] = @Original_CompanyName) AND ((@IsNull Contac
    "Name] IS NULL) OR ([ContactName] = @Original_ContactName))
    "Title = 1 AND [ContactTitle] IS NULL) OR ([ContactTitle] =
    "e)) AND ((@IsNull_Address = 1 AND [Address] IS NULL) OR ([
    "ddress)) AND ((@IsNull_City = 1 AND [City] IS NULL) OR ([C
    ") AND ((@IsNull_Region = 1 AND [Region] IS NULL) OR ([Regi
    ")) AND ((@IsNull_PostalCode = 1 AND [PostalCode] IS NULL)
    "iginal_PostalCode)) AND ((@IsNull_Country = 1 AND [Country
    "y] = @Original_Country)) AND ((@IsNull_Phone = 1 AND [Phon
    "] = @Original_Phone)) AND ((@IsNull_Fax = 1 AND [Fax] IS N
    "inal_Fax)))"
```

So What are Wizards Good For?

- Rapid Prototyping
- It depends on your situation
 - “One size does not fit all”
- Be aware of the trade-offs

Dataset Dilemmas

The Dataset is Awesome !!!

The Dataset is very powerful and is improved in .Net 2.0

- Disconnected data store
- Ability to associate with CRUD capability via DataAdapter and TableAdapter
- Searching
- Sorting
- Filtering
- Keeps track of Inserts, Updates, Deletes
- Serializable
- Can define table relationships and constraints

Dataset Issues

While many performance related issues have been addressed in .Net 2.0, several design related concerns must be considered ...

- Replication of Relational and Data Integrity Rules
- Tight Coupling with the Data Model
- Coordination of Work
- Validation and Business Rules
- Interoperability

Replication of Relational and Data Integrity Rules

The Dataset provides the means to mimic the functions of a Relational Database

- Referential Integrity

- **ForeignKeyConstraint object**

- **DataRelation object**

- **FillSchema method**

- Requires extra round-trip to database

- Other Data Integrity Options

- **PrimaryKey property**

- **UniqueConstraint array, Unique property**

- Doesn't check against rows not retrieved

Replication of Relational and Data Integrity Rules

- These rules should probably match those defined by the DBMS
 - If the rules are only built for Datasets, you're missing the power of the relational database
- **Much work may be required to keep these rules in sync**
- Should these rules be replicated ?
 - Navigation for Master-Detail scenarios?
 - **Business Objects provide an alternative**

Tight Coupling with the Data Model

- DataTables (typically) assume structure and type definitions of Tables or Views
 - **SELECT statements or Stored Procedures**
 - **FillSchema**
- Consumers (a.k.a. clients) may expect certain columns and data types
- If data model changes, other layers may throw exceptions
- Domain Objects provide a layer of abstraction
 - Also helps in versioning

Coordination of Work

- Referential Integrity rules in the data model may mandate ...
 - Step 1. Inserts, Updates, Deletes in parent tables
 - Step 2. Operations in child tables
- Where should this logic be orchestrated from when using Datasets?
 - Partial Classes ?
 - Stored Procedures ?
 - Domain Objects ?
 - Services ?

Validation and Business Rules

- Business rules may require that math or other operations be performed before CRUD
- Where do you put validation logic when using Datasets ?
 - Partial classes ?
 - **Good choice for TableAdapters**
 - Specialized validation classes ?
 - The UI ? (noooooooooooooo !!!!)
 - Behind the Service Façade ?
 - **Services should never trust the data they receive**
 - **Consider implications for when validation logic is distributed**

Datasets and SOA

The Class Conundrum

SOA Principles

- **“Services Share Schema and Contract, Not Class”**
 - Don Box
- An XML Schema Definition (XSD) = The Contract
 - WS-I rules mandate inclusion of schemas used in all operations
 - **WS-I Basic Profile, Rule R2001**
 - Still, may want to store schemas in referenceable location
- A Major Goal of SOA = Interoperability

Datasets and SOA

Datasets pose a problem for interoperability ...

- Datasets are .Net specific
 - This includes all sub-classes
 - **Typed Datasets, TableAdapters**
- WSDL for operations containing datasets do not provide explicit contracts until runtime
 - IsDataSet= True
 - **Hints that XML should be deserialized as a Dataset**
 - **Useless to Non .Net consumers**
 - Non .Net consumers have no clue about message structure
 - **Forced to parse a potentially changing structure**
 - **Remember that DataSets can morph their structure if Table definitions change**

Demonstration

Use of TableAdapter

- Wizard Generated
- Show how to use to generate XML messages for .Net and Non-.Net consumers from the TableAdapter
- Highlight
 - **XML Schema Definition (XSD)**
 - **WSDL for the Service that returns Dataset**

Implications

Datasets don't promote interoperability

- Can still use Datasets ...
 - Behind the Service façade
 - At the Consumer
- If you might have Non .Net Consumers
 - Don't pass Datasets back from Services
 - Forget about binary serialization of Datasets too

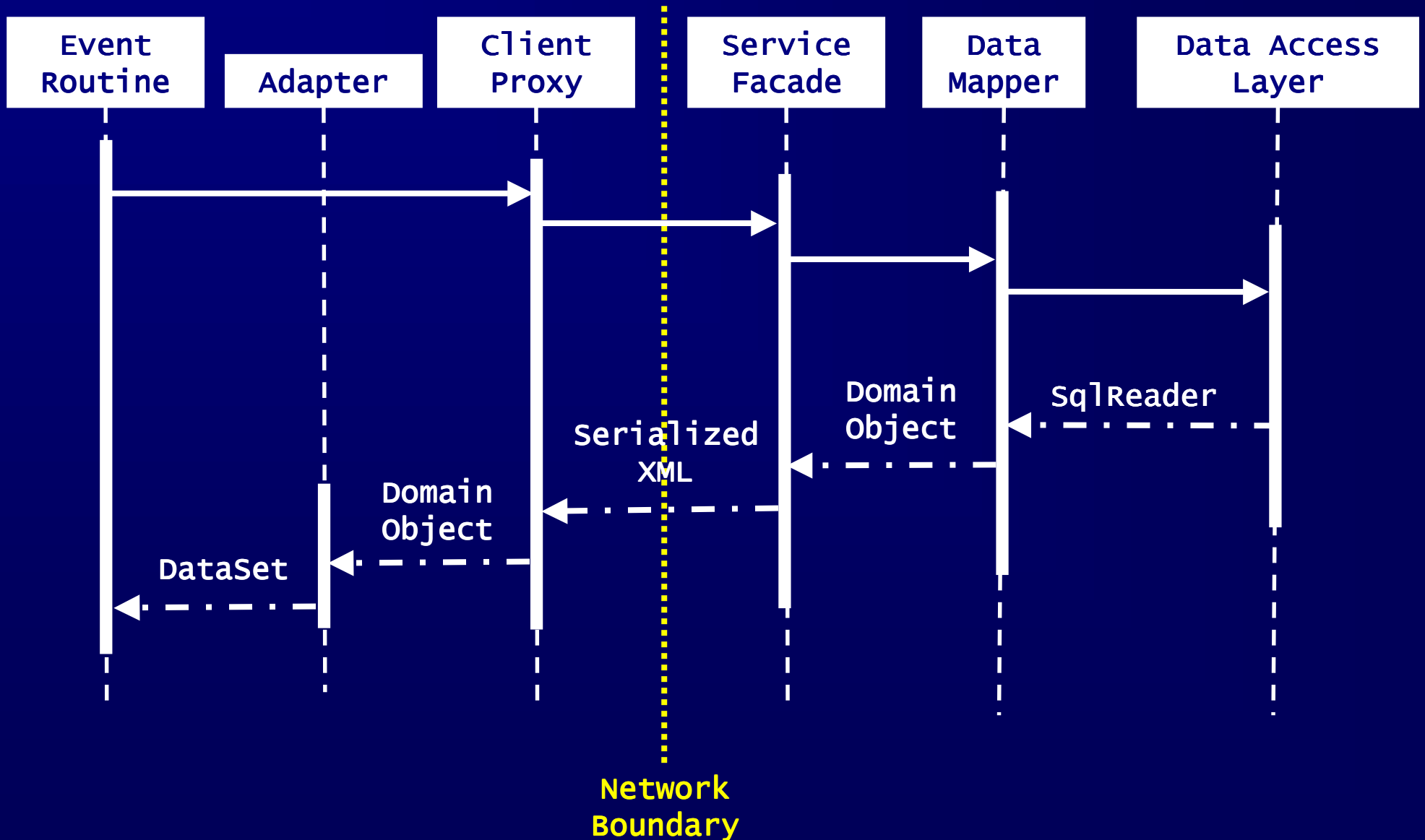
Other Dataset Problems

- XML Messages generated for Datasets are HUGE !!!
 - Schema
 - Diffgram
 - Relationships, Constraints, Keys
- Do NOT Publish Generated XML Schemas
 - Can provide clues for SQL Injection Attacks
 - **Data Source Name**
 - **Table, Proc names**
 - **Column names**

Demonstration

- Service that promotes interoperability
 - Use-Case Orientation (vs. Class Orientation)
- Use of Domain Objects behind Service Façade
 - Partial Classes to store logic for ...
 - **Data Mappers**
 - **Data Access Layer**
 - **Fields**
 - **Validations**
- Use of DataSets on a .Net Consumer
 - Receives XML message from Service
 - Tracks Inserts, Updates, Deletes

Demonstration Overview



Demonstration Recap

- Patterns demonstrated
 - Data Mapper / Object-Relational Mapper
 - Data Accessor
 - Remote Façade, Service Layer
 - Domain Object
 - Proxy / Broker
 - Adapter

You Can't Have it All

You can't have it all ...

Datasets, Speed of
Development

Centralized Data Access
Logic, Ease of Maintenance



Datasets, Speed of
Development

Flexibility, Loose
Coupling with Data Model



Datasets, Speed of
Development

Interoperability



Recommended Resources

- www.DesignPatternsFor.Net
- Patterns of Enterprise Application Architecture
 - Martin Fowler , Addison Wesley 2003
- Data Access Patterns
 - Clifton Nock, Addison Wesley 2003
 - **Written for Java developers**
- Expert Service-Oriented Architecture in C#
 - Jeffrey Hasan, Apress 2004