

.Net Training

ORM (C#)
(Object Relational Mapping)

Agenda

- Goals of Object-Oriented Design
- Business Objects vs. DataSets
- Object Relational Mapping Overview
- ORM Strengths
- ORM Critique
- NetTiers



Goals of Object-Oriented Design

- To support a fully object-oriented programming model
- To allow the developer to use the architecture without jumping through hoops
- To enable high scalability
- To enable high performance
- N-Level undo on a per-object basis (Edit, Cancel, Apply)
- Tracking of broken business rules
- Support for many types of UI based on the same business objects
- Integration with transaction engine (MTS/COM+)
- Simplify complex issues like serialization and remoting
- Be able to use VS.Net tools like Intellisense and Autocompletion

Business Objects vs. DataSet

- **Undeniable Advantages of Business Objects vs. DataSets:**
- DataSets are Microsoft specific
- If you need to pass them into a non-Microsoft, i.e. Java web service you will need to create a separate business object to pass to it
- In large organizations, technology choices are like a light switch; one IT manager comes in and loves .Net and the next manager comes in and switches the company standard to Java

Business Objects vs. DataSet

- Be ready to support both legacy and future switches by having generic business objects
- Business objects allow business rule validation before data is saved to the database
- Nothing is worse than having users upset due to runtime errors resulting from bad imported data
- Business objects have better encapsulation and extensibility than DataSets
- Code reuse lowers project costs

Business Objects vs. DataSet

- Code generators such as CodeSmith, MM.NET, MyGeneration, etc. can generate all the persistence layers for an application
- They actually generate more capabilities than what is built into DataSets
- The templates that generate the output can be customized – not so with DataSets
- The code generators also allow regeneration when the database schema changes – this is more difficult in DataSets

Business Objects vs. DataSet

- DataSets almost beg the developer to place the business logic in the user interface layer
- Having a business rules region in a business object woos the coder into placing the business rules in the proper place
- This lowers the cost of putting a different user interface on the application – reusability!
- Using a DataReader to populate Business Objects has up to 30x more performance than using a TableAdapter to fill a DataSet

Business Objects vs. DataSet

- Business objects are easier to regression test
- Having business rules spread between the database, partial classes for a DataSet, and the user interface layer makes it much more difficult to regression test
- It's easy to miss defects, unless expensive high end user interface testing tools such as WinRunner or Rational Robot are used

Business Objects vs. DataSet

- Testing frameworks such as NUnit or Microsoft Test allow a high quality product to reach the end user, when using business objects
- When considering DataSets vs. business objects, remember that a well designed application using business objects = less future headaches in the maintenance = lower support costs

Object Relational Mapping

- One of the disadvantages of using Business Objects versus DataSets is the extra goodies bundled in the DataSet that facilitate databinding
- Without those the programmer has to implement those interfaces him/herself
- This is where ORM comes in
- Using a template-based approach vast amounts of object-database code can be generated using ORM techniques in a short amount of time using ORM code generators

Object Relational Mapping

- ORM – Object Relational Mapping
- Is a programming technique for converting data between incompatible type systems in databases and Object-oriented programming languages
- This creates, in effect, a "virtual object database" which can be used from within the programming language
- There are both free and commercial packages available that perform object-relational mapping, although some programmers opt to create their own ORM tools

What is an ORM tool?

- Object-Relational Mapping
 - Mapping database fields to objects
- Code generation
 - Basic CRUD
- RAD – Rapid Application Development
- Standardizing Data Access code
- Reducing bugs
- Time-to-Market
- Cost-Benefit

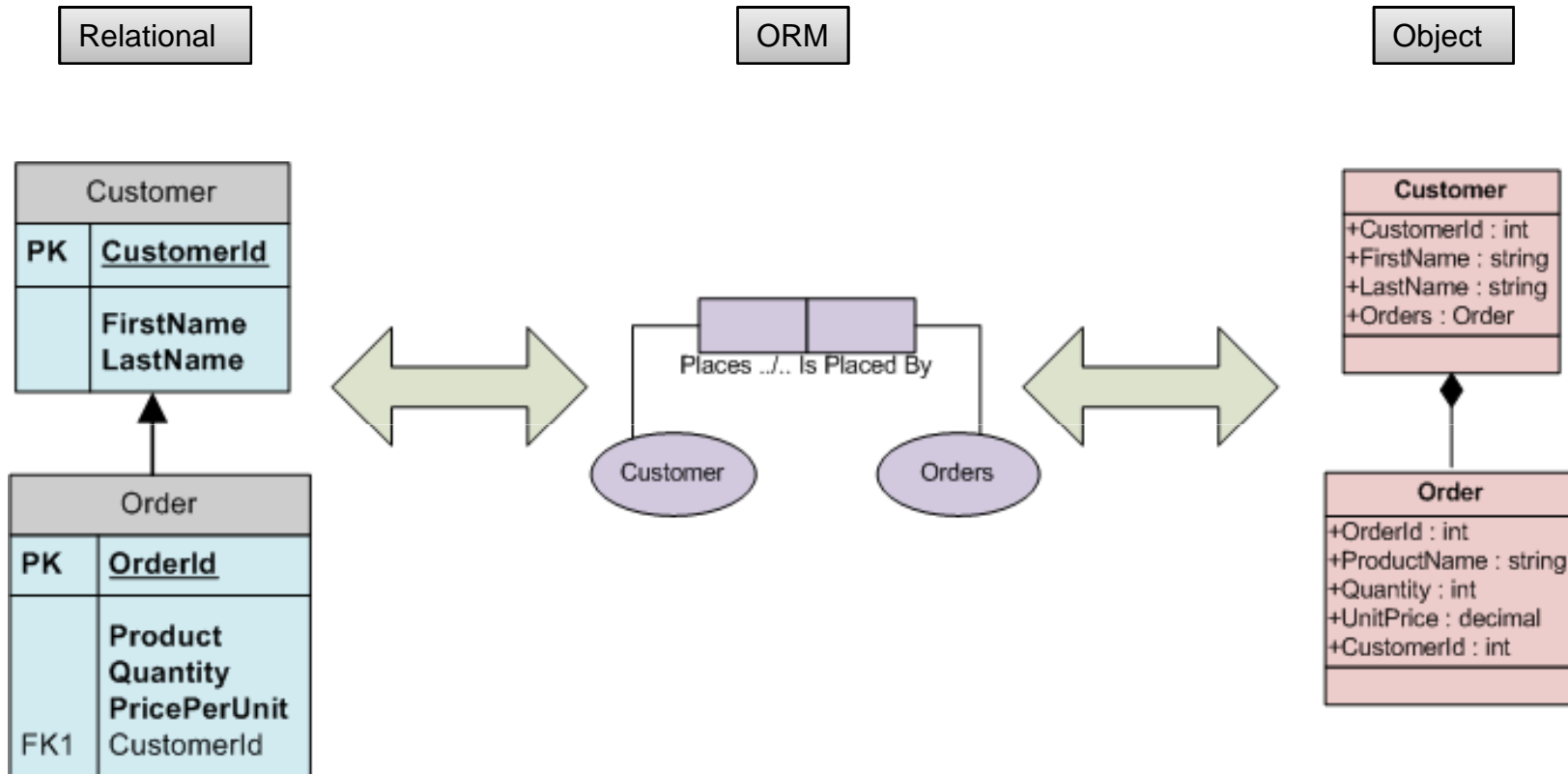
ORM Strengths

- Facilitates rapid business logic development, by taking care of typically time-consuming and repetitive tasks
- Speeds up time-to-market in new developments by about 70% compared to VB6 platform
- Speeds up feature enhancement/bug fixing by about 75% compared to VB6 platform
- Less probability of bugs occurring by encapsulating reusable business objects.
- More time spent developing new features, less time spent on infrastructure/bugs
- Increased security
- Increased performance
- More flexibility

ORM Strengths

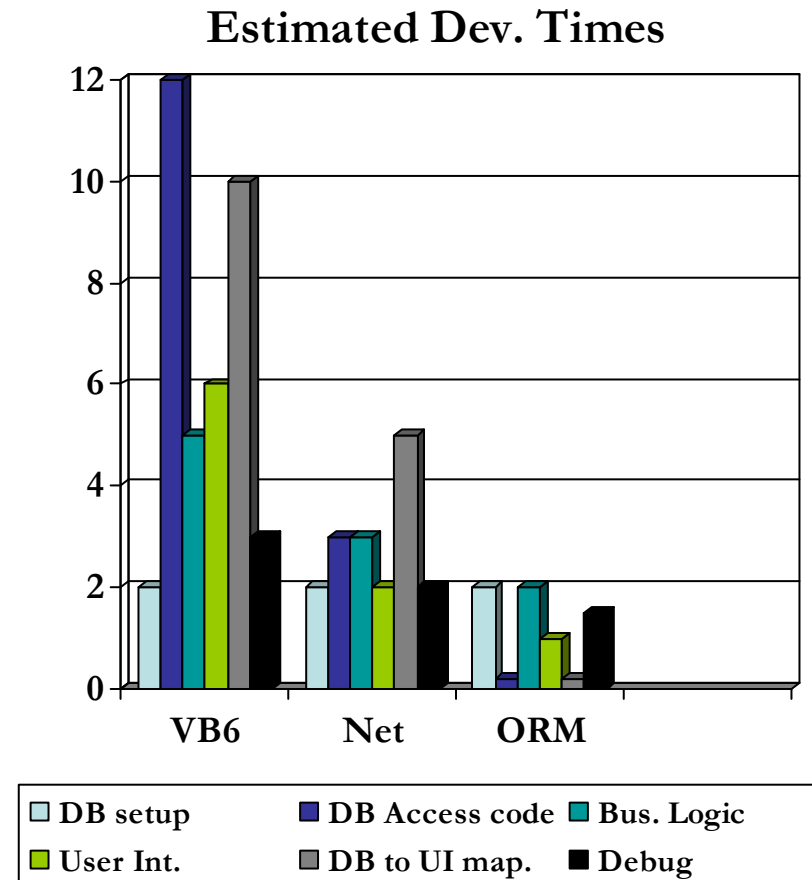
- Fewer skill sets required by new Developers
- Faster to code with smaller staff
- Less time to train new developers
- Ability to support new interfaces, from Web Services, to ASP.Net, Windows clients, to potentially Pocket PC's and other PDA's – all from the same re-usable business logic
- Ability to provide more customizations and support those customizations more easily
- Increased employee satisfaction
- Ability to focus more on our core business and worry less about code issues
- Reduced costs

Object Relational Mapping



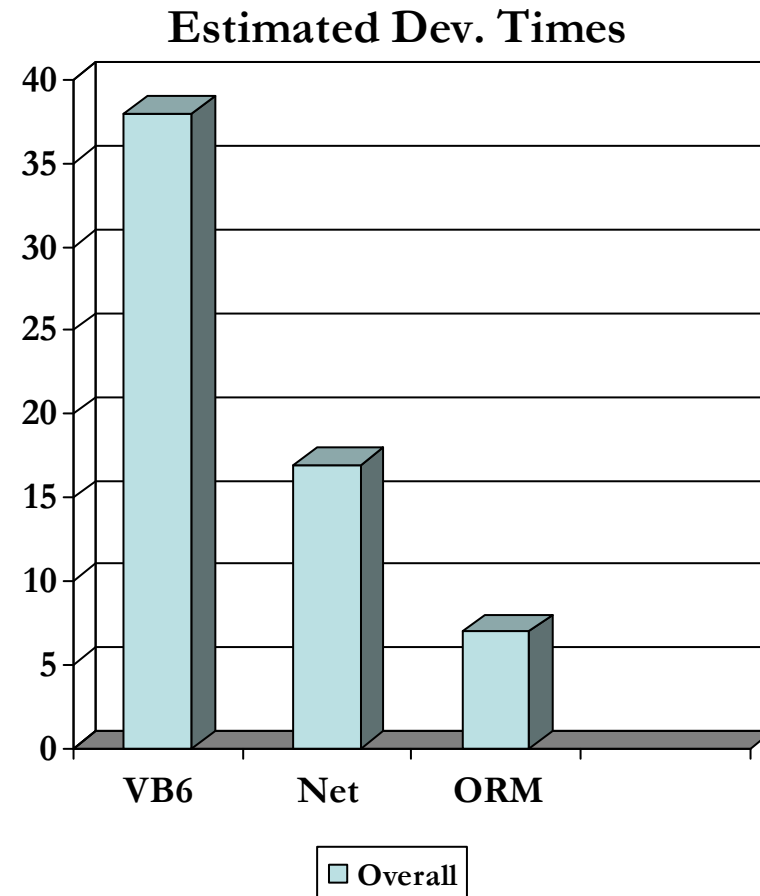
Dev. Life-Cycle Comparison (Windows)

- Considering the same Windows application, just moving to .Net provides several advantages over the VB6 platform, in terms of time-to-market (usually by about 55%)
- An ORM tool increases that factor by another 30-40% overall, by greatly reducing the amount of time developers spend on wiring database access logic to business logic and then to UI fields



Dev. Life-Cycle Summary (Windows)

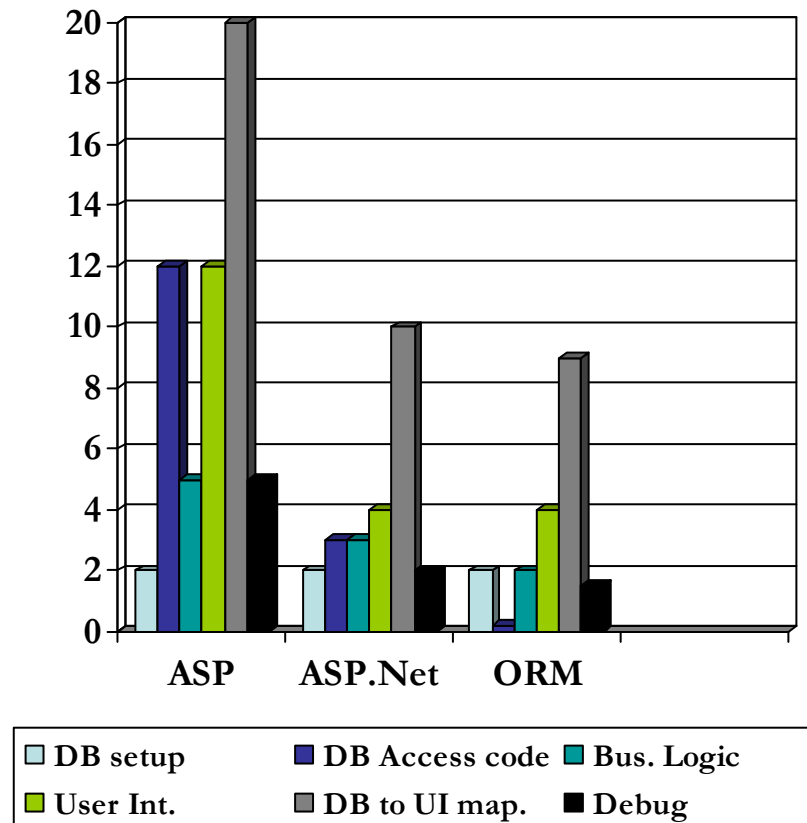
- Overall we estimate a combined 70% increase in productivity and time-to-market moving to .Net and an ORM tool versus VB6



Dev. Life-Cycle Comparison (Web)

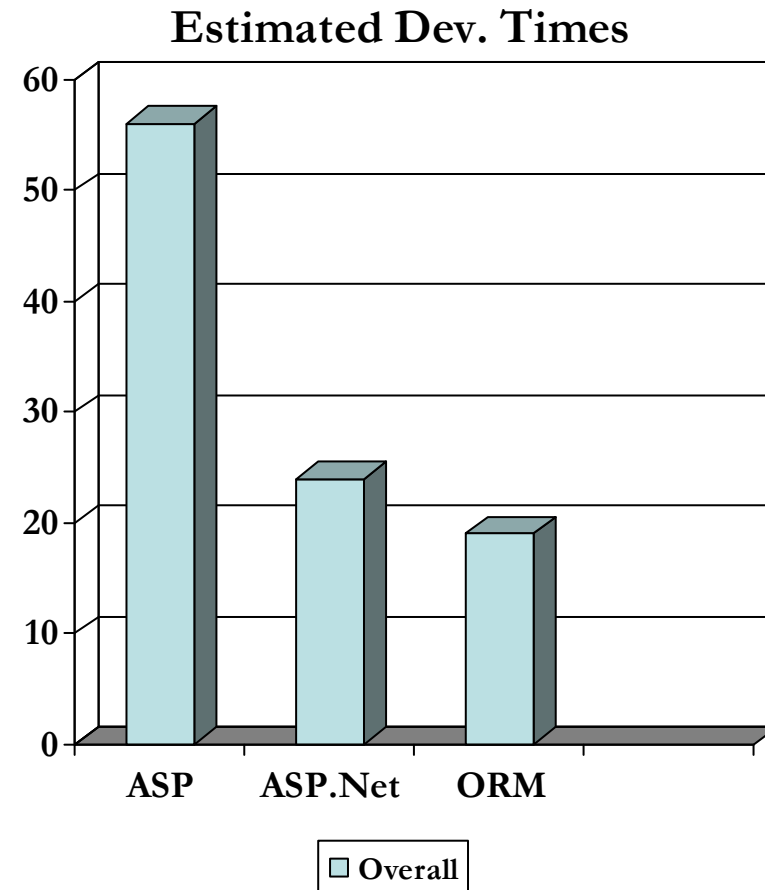
- Considering the same Web application, just moving to .Net provides several advantages over the ASP/VB6 platform, in terms of time-to-market (usually by about 57%)
- An ORM tool increases that factor by another 20% overall, by greatly reducing the amount of time developers spend on creating database access logic and wiring database fields to business logic and UI fields

Estimated Dev. Times



Dev. Life-Cycle Summary (Web)

- Overall we estimate a combined 65% increase in productivity and time-to-market moving to .Net and an ORM tool from classic ASP



ORM Critique

- Some say the promotion of ORM is an attempt to handle the wrong side of the problem
- Critics propose that instead of ORM we should be focusing on solving the problems with relational databases and make them more object-oriented
- This would make ORM redundant and eliminate most of the tedious work out of coding
- A number of Object Databases have surfaced
- But these have not gained traction in the market

Object Relational Mapping

- There are many .Net ORM tools out there:
 - IdeaBlade
 - ORM.Net
 - Constructor.Net
 - NetTiers
 - Wilson's O/R Mapper (WORM)
 - DataObjects.Net
 - eXpress Persistent Objects for .Net
 - Objectz.Net
 - Etc.

Object Relational Mapping

- But ORM is not an exclusive .Net concept:
 - NeXT's Enterprise Objects Framework (then NeXT's WebObjects, now Apple's Objective-C and WebObjects for Pure Java)
 - Apache Cayenne
 - SPARQL
 - Java Data Objects
 - EJB 3.0
 - Ruby On Rails
 - Python's SQLAlchemy, SQLObject and Dejavu
 - Etc.

Object Relational Mapping

- Similarly, many competing Business Object frameworks have emerged, both within and without the .Net arena
- Some involve ORM, some do not
- One of the most respected frameworks emerged from the mind of Rockford Lhotka, a Microsoft Software Legend, Regional Director, MVP and INETA speaker – CSLA.Net (formerly CLSA for COM/COM+)
- NetTiers is based on portions of CSLA.Net

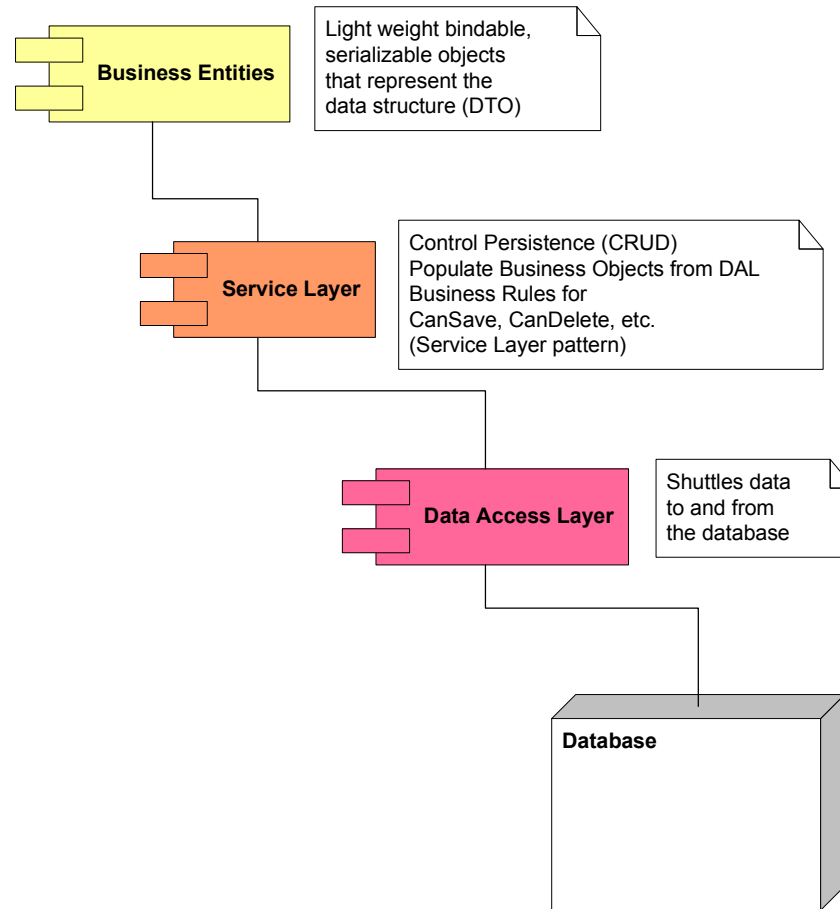
NetTiers

- NetTiers is a ORM framework that is used via CodeSmith templates to map relational database tables and views to Business Objects implemented with the CSLA.Net philosophy
- NetTiers is free
- NetTiers renders all the code without 3rd party dependencies
- There are no hidden parts – you have a access to all the source code
- You can debug through all the parts

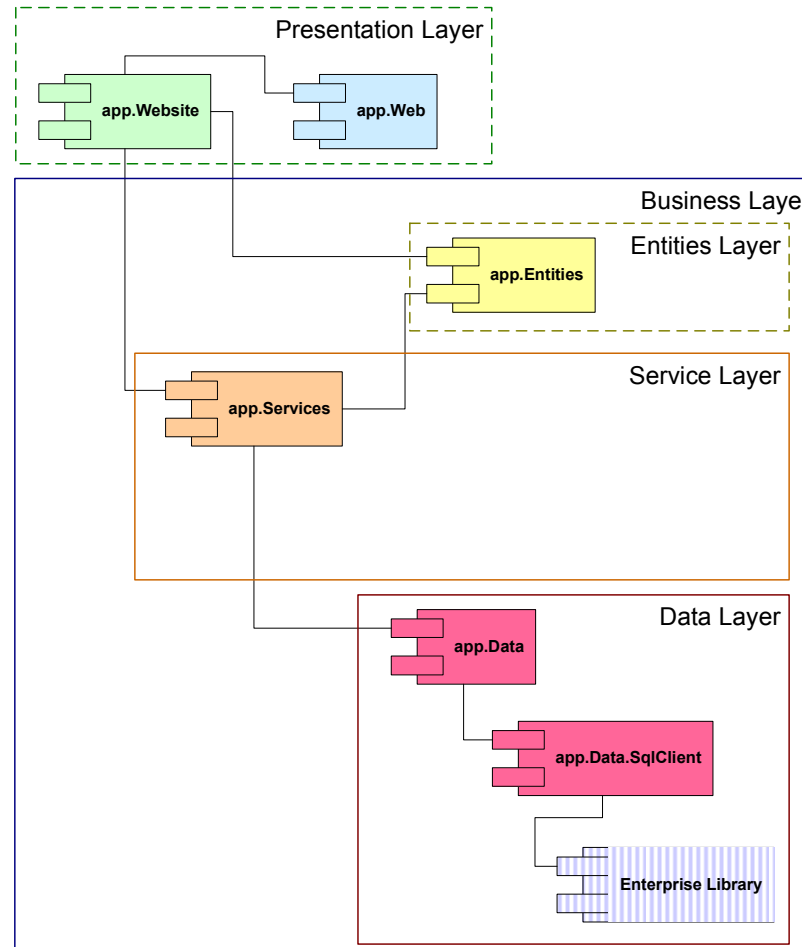
NetTiers

- ORM tool (Object-Relational Mapping)
- Code generation
- Template based
- Uses CodeSmith
- Uses Microsoft Enterprise Library
- Uses Design Patterns

NetTiers Architecture Overview

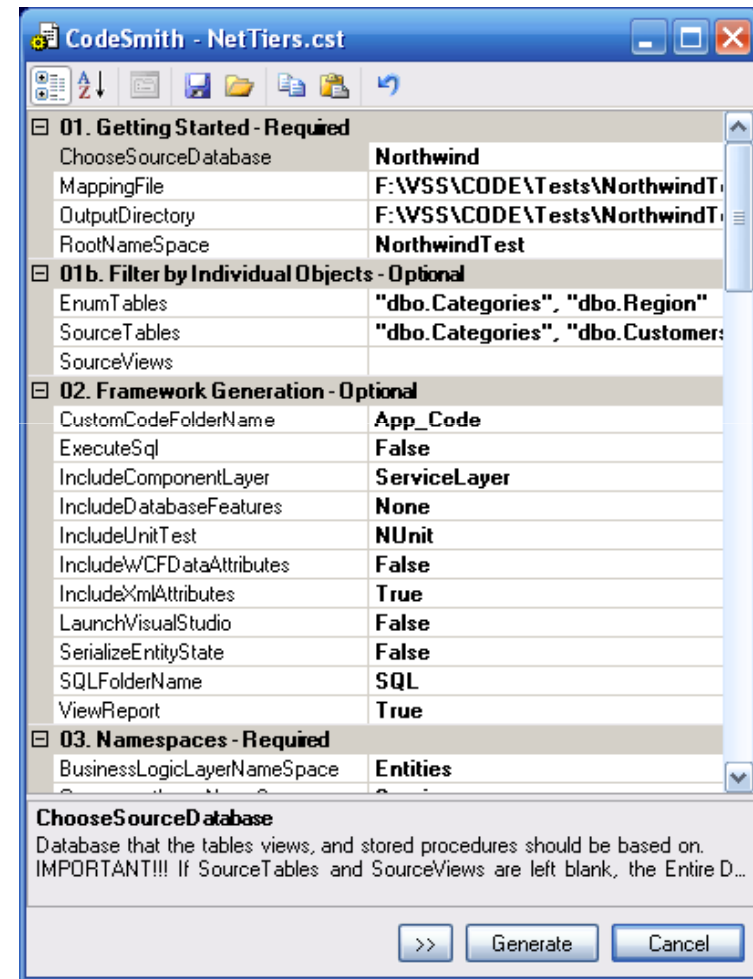


NetTiers Architecture Overview



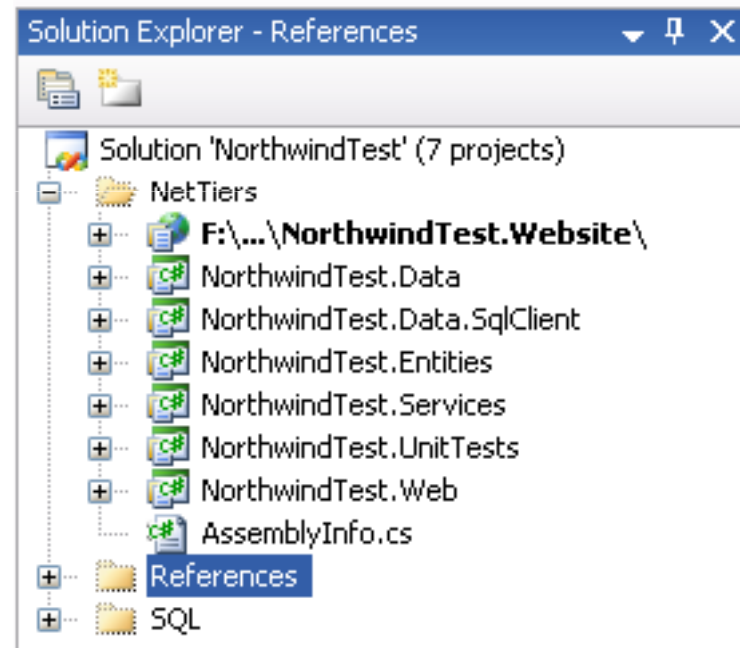
NetTiers Process

- Open NetTiers template with CodeSmith
- Point to Database
- Set Namespace and path
- Select Tables/Views
- Set Properties
- Click Generate
- Open code in Visual Studio

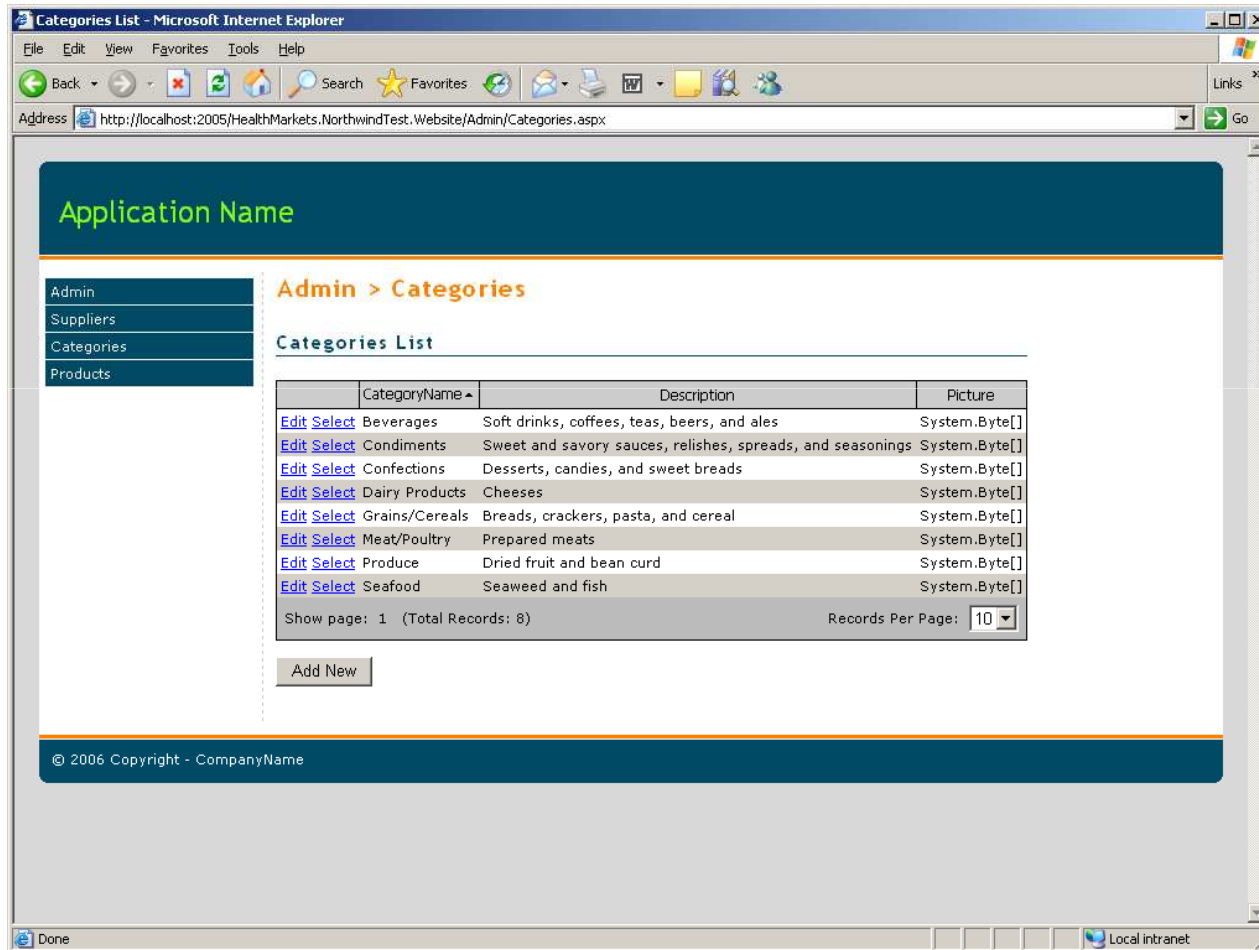


What you get

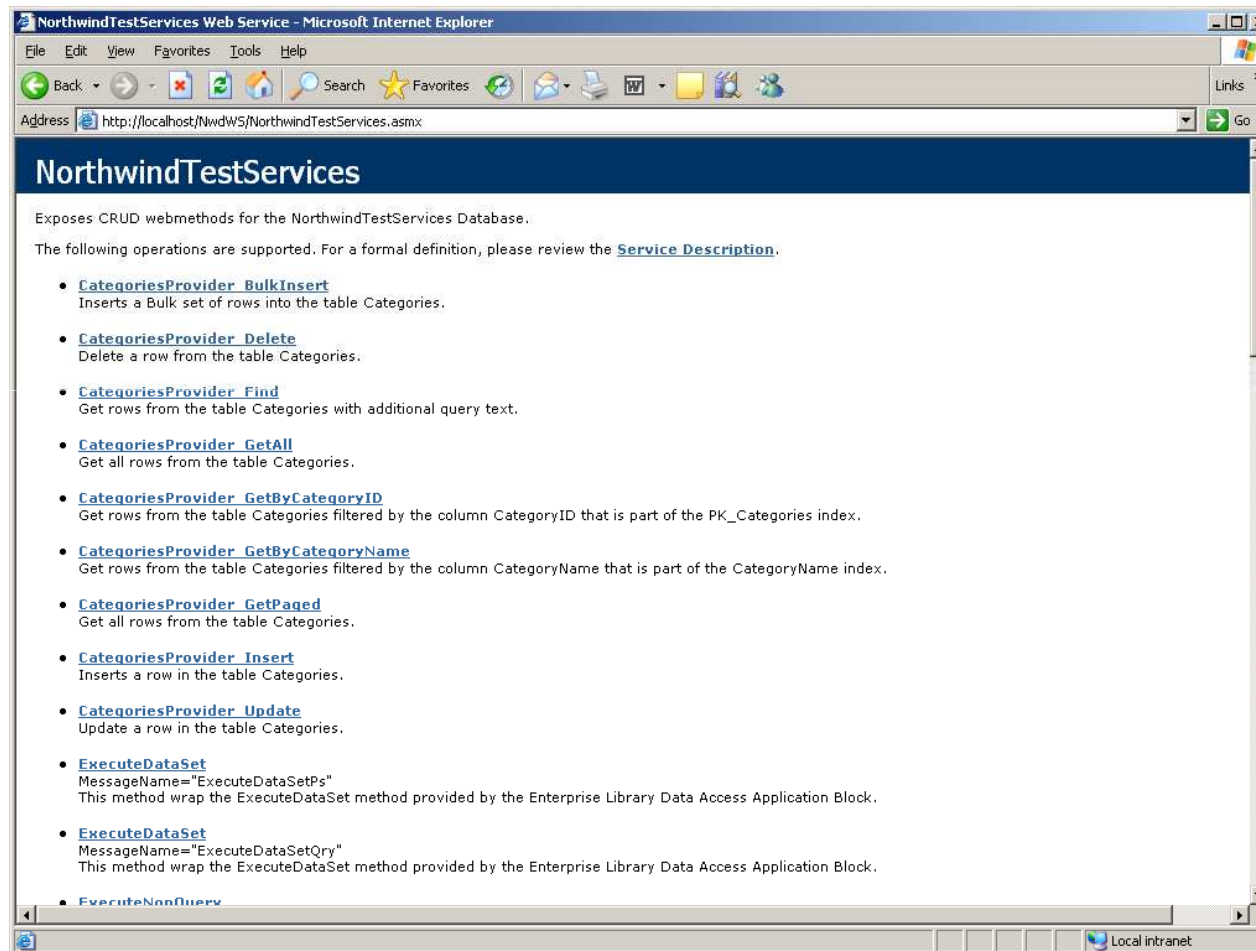
- N-Tiered solution
- Entities Layer (business objects)
- Data Layer
- Services Layer
- Stored Procedures
- Unit Tests
- References
- Documentation
- ASP.Net Web Utilities
- ASP.Net “Scaffolding”
- Web Services
- WCF
- AJAX (Atlas)



What you get



What you get



NetTiers Advantages

- Made by Developers for Developers
- Full control of template and code generation - Customizable
- Open Standards
- Industry-Wide accepted
- Based on Design Patterns
- Free
- Support available
- Saves time and effort (time to market)
- Objects vs. Datasets
- Easy to regenerate when database changes
- NUnit or TFS unit testing integration
- Power of the Microsoft Enterprise Library
- Stored Procedure based
- Codesmith has other uses
- Enum Classes!

NetTiers Disadvantages

- Only works with SQL Server databases (currently)
- Issues with Source Control
- Minor quirks require a little tweaking
- Staff training
- Databases have to be properly normalized for the most features to work

Further Reading

- <http://docs.nettiers.com>
- <http://community.codesmithtools.com/blogs/videotutorials/archive/2006/02/07/nettiers.aspx>
- <http://www.lhotka.net/Area.aspx?id=4#>
- <http://www.howtoselectguides.com/dotnet/ormapping/>