

Object-Relational Mapping Unveiled: Tangent Systems Proves Efficiency of ORM with Visual Studio and Vanatec OpenAccess Union

Learn how six months of benchmarking and test programming to find the right object-relational mapping tool can pay off. Vanatec's OpenAccess turned out to be just what Tangent Systems was looking for to allow its customers to process a wide range of documents quickly, efficiently, and easily.

by Shari L. Gould
February 1, 2007

The decision to migrate the company's code base from C to an object-oriented (OO) language, along with the ability to define the numerous databases, was not an easy one for [Tangent Systems](#), Inc.'s founder Steve Mack. In fact, the company's high-speed, high-volume imaging solutions operated with C code for 21 years before Mack and his team made the leap. Microsoft's C# and .NET platform provided the cohesive OO solution they wanted, but the question still remained regarding how to handle persisting objects to a SQL database. Should they write conventional data mappers to do the job, or use object-relational mapping (ORM)?

The decision came after arduous research and trials of hand-crafted data mappers and the object-relational mapping tools available, conducted by Darrell White, a Senior Software Developer with Tangent. Tangent's development team ultimately chose the ORM approach. Here's why:

Traditional data mappers create data element mappings between two distinct data models, but this approach is not suitable for OO applications. Data mapping is used as a first step for a wide variety of data integration tasks, such as data transformation or data mediation between a data source and a destination; identifying data relationships in data lineage analysis; discovering hidden, sensitive data during data masking; or consolidating multiple databases into a single database. However, traditional data mappers do not provide persistence for the .NET platform, C# applications, and custom business objects.

A conventional OO data mapper layer mitigates the shortcomings of a relational database in persisting the varied data structures and relationships of objects. The data mapper layer also provides some isolation between the object world and the database world, so that changes in one do not ripple directly into the other. However, data mapper code can consume a significant portion of the development time for an OO project, since mapping objects to relational tables can be cumbersome.

On the other hand, the ORM programming approach links databases to objects to create a virtual object database. As a result, developers should not need to write data mappers, scripts or any SQL code to create and map objects to database tables. You also should not have to track your changes to your objects. It should all be done seamlessly for you with a comprehensive ORM solution.

Test Driving

White's six-month process of consulting experts and test driving a series of tools led to frustration and the conclusion that most ORM tools left much to be desired and simply did not provide the functionality Tangent Systems needed. He created rigorous test programs and thorough benchmarking uncovered deficiencies—such as in functionality, performance, and the lack of seamless persistence—in most ORM solutions.

As a company that provides high-speed document processing software—primarily to financial industries—Tangent couples the efficiency of high speed document readers with software to provide image display and workflow control. The goal is to allow its customers to process a wide range of documents quickly, efficiently, and easily.

Enter OpenAccess

After six months of benchmarking and test programming, White put Vanatec's OpenAccess [<http://www.vanatec.com>] ORM solution to the test. White created a test class and created an ORM helper that worked with OpenAccess and their SQL databases. The result was a smooth application that talked directly to the SQL databases without the expertise of a database administrator, which the company does not employ. OpenAccess provided transparency. White said they don't even know the code is there that creates database tables, generates scripts, and tracks database changes, using low-level persistence services. OpenAccess provides more efficient persistence than if White had written his own mapper, and it is far better than a manually configured ORM product.

Next, he created an enhanced ORM helper that disconnected collections simply by changing parameters in a configuration file, making the application talk to a Web service. The ORM helper White developed were ultimately used in the company's first mission-critical OO project using OpenAccess, Visual Studio, and C#, called Deposit21. This large, database-intensive application performs remote electronic deposits using various industry-standard formats and methods, including X9.37, X9.100-180, and other bank-specific variations. Besides image deposits, Deposit21 is also ARC-capable.

Delivered within a tight timeframe to meet customer requirements, OpenAccess, used in conjunction with Visual Studio, saved Tangent Systems 60 percent in total development time, said Mack. In terms of lines of code, White estimates that OpenAccess generated code on the fly that would have needed a DBA to write at a factor of 100 to one. Further, OpenAccess substantially impacted the database schema changes, because it automatically finds changes in the objects and updates them in the database schema. Mack said that they have not yet used Open Access's reverse mapping feature, since they are persisting objects and not accessing legacy databases, but the feature is an important one for future requirements.

White said one reason the company chose Vanatec's OpenAccess ORM solution is its integration within Microsoft's Visual Studio, easily accessed from a menu directly within Visual Studio. Further, no other ORM product that was tested had the object container concept like Vanatec's OpenAccess, where the ObjectContainer class is used to share persistent objects with distributed processes that do not have a database connection.

OpenAccess allows developers to seamlessly participate in the persistence process using containers. Persistence, in this context, means objects will be stored in the database in between sessions of the application. All fields are Persistent unless they are marked with the Transient attribute.

How OpenAccess Works

OpenAccess is an object-relational mapper providing transparent persistence that fully conforms to the .NET standard. OpenAccess allows you to easily store your .NET application objects directly in the underlying database system. OpenAccess provides developers with a smart data

access layer, which provides persistence services for effectively building well designed object-oriented applications. The custom business objects form an object model where business logic can be written in terms of the properties and methods of objects in the object model instead using SQL syntax. OpenAccess handles the mechanics of translating changes to your objects at runtime to SQL statements that get executed against the database.

OpenAccess requires you to provide metadata about your business objects using .NET attributes. You may also describe inheritance relationships and associations between your domain classes using .NET attributes. OpenAccess' use of metadata that describes your objects allows you to query for persistent objects using Object Query Language (OQL), which has been defined and standardized by the Object Data Management Group (ODMG) instead of writing SQL queries (which is nevertheless still possible). OpenAccess translates the object query into a SQL query on the backend database and returns objects to your application. The future additional query language will be LINQ, currently implemented as preview support based on the current CTP. Also, as your application modifies the state of these persistent objects, OpenAccess invisibly tracks the changes. When you tell it to, OpenAccess writes the changes back to persistent storage without you having to write a single SQL statement. As it writes the changes, it also intelligently handles object relationships.

Final Note

Eliminating the mismatch between an object and a relational model increases developer productivity, something to which Tangent Systems can testify. So, companies with database intensive applications that are considering migrating their applications don't need to become database experts to close the gap between objects and relational models. ORM solutions like Vanatec's OpenAccess do the work for you.

While object persistence is a key advantage of ORM tools like Vanatec's OpenAccess, it also provides many advanced features for building enterprise applications. They include:

- Flexible mapping options between objects and table schema
- Caching mechanisms for objects to minimize expensive trips to the database
- Lazy loading of objects; a disconnected model allowing objects to be disconnected from the domain model and shipped to another process-enabling Application Server, Web and mobile applications
- An Object Query Language to query for business objects
- Databinding to standard .NET controls such as the DataGrid; support for building Windows Forms, Web Forms and Web Services applications
- High Performance Optimization through Fetch Plans
- Full integration into the Visual Studio .NET IDE with Intellisense support
- Preview support for the current LINQ CTP
- Fully integrated documentation.

More Resources:

- [Download: Vanatec OpenAccess Express](#)
(full license for free databases such as Microsoft SQL Server 2005 Express)
- [Download a free trial of Vanatec OpenAccess](#)
(30-day license without any limitations)
- Whitepaper: [Introduction to ORM and Vanatec OpenAccess](#)
- Find out more about object-relational mapping at www.vanatec.com

Shari L. Gould has more than 16 years of journalism and technical writing experience. Shari has written for numerous leading publications throughout her career, most recently Software Development Times and its various publications, and had an article hand picked by Sun Microsystems for inclusion in its Solaris Developer

Connection. She also has more than 10 years experience working with high-tech companies documenting everything from network designs and installations, through software design and APIs, to user interfaces. Shari currently is pursuing her Master's degree in Criminal Justice, specializing in Information Security.

© 2007 Microsoft Corporation. All rights reserved.