ODBMS.ORG User Report No.4/08

Editor Roberto V. Zicari- ODBMS.ORG www.odbms.org

July 2008.

Category: **Industry**
Domain: Various
User Name: **William W. Westlake**
Title: Principal Systems Engineer
Organization: ***Science Applications International Corporation*, USA**

Bill Westlake is a senior architect and engineer with extensive experience ranging from web application development to military vehicles. He is an electrical engineer by education with extensive experience in electronics from communication to hand held computing devices. He is an experienced software developer and architect with with experience using Together Soft products, Rational Rose, ClearCase, ClearQuest, RequisitePro, Microsft Visual Studio 6 and Visual Studio .NET.

Q1. Please explain briefly what are your application domains and your role in the enterprise.

William W. Westlake : As a systems engineer it is my job to look across the entire system requirements and identify technical solutions that can be applied to the entire system eliminating duplication. In performing this function we have often been confronted with multiple data storage systems for different domains within a system of systems. Because of this I have often applied object persistence as a remedy, reducing both the cost of development, as well as the long term maintenance costs.

Q2. When the data models used to persistently store data (whether file systems or database management systems) and the data models used to write programs against the data (C++, Smalltalk, Visual Basic, Java, C#) are different, this is referred to as the "impedance mismatch" problem. Do you have an "impedance mismatch" problem?

William W. Westlake : I have done software development at a variety of companies that suffered from this issue. One of them was a medical data warehouse. This warehouse collected data from patients that participated in various medical studies. The data sources varied greatly in the data format.

The solution to this issue in these cases was to build an object persistence system, in this case using Enterprise Java Beans and CloudScape. All incoming data was then transformed into this format and persisted. Data could them be extracted back out to formats that customers requested for their research.

Q3. What solution(s) do you use for storing and managing persistence objects? What experience do you have in using the various options available for persistence for new projects? What are the lessons learned in using such solution(s)?

William W. Westlake : My first experience with using object persistence was with Enterprise Java Beans. In my experience EJB's required too much discipline and too many skill sets within an organization. Development skills weren't the issue, but rather the packaging and deployment skills. The IT group required a lot of training, and new processes had to be established.

Object persistence that requires less management overhead, less database upkeep, and more easily validated objects are the best solution. We discovered this with MS .NET and Db4objects. Between the two we have been able to roll out new solutions much more easily with little extra training.

We used MS .NET and Db4objects to develop a catalog for a medical information database. The medical information that we managed is used by medical researchers and is tightly controlled. In order to limit access to the details stored in an Oracle database we used Db4objects to create a catalog that updated when ever new information was placed in the Oracle DB. When an order for information came in the people processing the order used the catalog database reducing load on the Oracle DB.

The system would then automatically process the orders and package the data for delivery. We did not have to develop a schema in Db4objects, just the objects we wanted to store which greatly reduced the development time. Since there is no special packaging and deployment mechanism -- as in EJB's -- we just had to test and deploy DLL's as the team developed them. The process we established for this worked very smoothly and allowed the IT department to quickly validate each assembly, and after several iterations of this they became very confident of the process.

The technical advantages are that the C# code describing each object is also the database schema. Modifications to that schema are easily made and managed, and the impact of proposed changes is easily assessed, because you have a single place where the schema and objects are described.

Contrasted with EJB's and object relational mapping tools where any changes to the EJB code base usually requires generating new underlying code and if not carefully managed across the various people in the different areas -- the coders, the database managers, the SQL developers, the testers, and deployers -- many difficult errors can accumulate.   Because Db4objects builds an internal schema based on the objects being stored this process is reduced to the coders and testers, and perhaps some IT people to deploy the code was validated. Few groups being involved made for less errors in deployment, few coding issues, and a much smoother project than others I have worked on.

Q4. Do you believe that Object Database systems are a suitable solution to the "object persistence" problem? If yes why? If not, why?

William W. Westlake: Absolutely, Object Databases are the best route to object persistence. With a RDBMS system you have to build and validate a schema, then you have to map the objects to the schema, then you need to properly deploy the objects. Any changes that effect the object data have a ripple effect, and in an RDBMS system this can be brutal. With an object database, such as Db4objects, you don't need to manage the schema at all, and with object versioning you have roll out updates over time with minimal impact, allowing the new version to work alongside the old version.

Q5. What would you wish as new research/development in the area of Object Persistence in the next 12-24 months?

William W. Westlake : I would like to see a true configuration management system for objects incorporated into the database. Extending object versioning to complete baselines of objects for a specified version. I would also like to see this be tied back to the source code version control system so that we can have assurance that the code and objects are in alignment and traceability to changes exists.

##