ODBMS.ORG User Report No. 33/10
Editor Roberto V. Zicari- ODBMS.ORG *www.odbms.org*
September 2010.

Category: **Academia**
Domain: **Research**
User Name: **Tilmann Zäschke**
Title: *PhD Student*
Organization: **ETH, Switzerland.**

Previously:
Category: **Industry**
Domain: **Aerospace**
Title: *Software Engineer*
Organization: **VEGA Deutschland, Germany /**
European Space Agency, Netherlands.

After graduating as an Engineer at the University of Darmstadt, Germany, Tilmann Zäschke worked on several projects for the European Space Agency (www.esa.int), hired as a contractor from VEGA Deutschland (www.vega.de). His last project there was the implementation of an persistence layer for the Herschel Space Observatory (www.esa.int/herschel).
In 2009, he became a PhD student in the Global Information Systems Group (www.globis.ethz.ch) at ETH Zurich, Switzerland, where his research focuses on agile development and object databases. In particular he investigates agile development and schema evolution in ODBMS which relates to his experience in agile projects in industry.

Q1. When you worked for the European Space Agency, what were your application domains and your role in the enterprise?

Tilmann Zäschke: My task was to realize a persistence backend for the Herschel Space Observatory. The Herschel Space Observatory is a satellite that performs observations in the far infrared spectrum, in particular observing very old objects with a high red-shift. The life time of the satellite is limited to 3-4 years, during which it is expected to produce 15TB of data.
The realization included several tasks: The main task was to design a strategy for making the satellite control data and raw observational data (refined data was handled elsewhere) available to the numerous involved parties with their different requirements. The solution included a custom hierarchical replication system to provide power-users with local data stores. Another problem was to provide the other developers (which did not have database

proficiency) with an easy way to access data, and hiding data localization and routing in a database system with many individual databases. Furthermore, the agile character of the project resulted in a continuous stream of schema changes, which had to be applied frequently to already existing customer data, which in turn made it necessary to develop a solution that was easy to use and least disruptive.

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. Did you have an "impedance mismatch" problem at ESA? And if yes, how did you manage it?

Tilmann Zäschke: Thanks to using an ODBMS we did not have any classic impedance mismatch. I have little experience with O/R mapping, but I believe if we had to maintain one, the frequent schema evolution would have become more than just a side task for me.
We still had two types of mapping to handle, first the mapping from the continuously evolving problem domain to the implementation / data model, and second improving the data model to make most of the chosen persistence solution. The latter means that we sometimes deviated from classic design policies in order to improve performance and accommodate characteristics of the persistence solution. For example we started introducing additional links to speed up and essentially avoid join-type queries.

Q3. What solution(s) did you use for storing and managing persistence objects?

Tilmann Zäschke: We used the Versant ODBMS 6.0 and 7.0 via the Java Versant Interface (JVI). As far as we could test it, the ODBMS should scale well with the 15 TB and 2 billion objects that we expect. However, we improved the available features to better suit the needs of our project. Since we decided to spread the data over multiple database node, we tried to make existence of multiple nodes more transparent to users developers and administrators. For example we decided to implement a custom replication mechanism for database systems (a system being a set of up to 50 individual databases), we implemented a simplified API for other developers which also managed transparent routing of

new data to the correct database node within a system. We also implemented some command line tools to simplify administration of database systems, including creation, schema initialization and user access management.
For schema and data evolution of whole database systems we introduced a set of applications for end-users to simplify schema evolution and for developers to auto-generate and verify schema and data evolution scripts.

Q4. What experience do you have in using the various options available for persistence for new projects?

Tilmann Zäschke: My main experience lies with Versant. The Herschel project also contains some JDO because the persistence API for other developers was roughly based on that standard. In an earlier project I worked with Ontos, which got replaced by Versant due to stability problems. Currently I am experimenting a bit with JDO and db4o and also with persistence solutions for OMS/Avon, a system that supports type evolution at runtime, which has been developed at ETH Zurich.

Q5. What are the lessons learned in using such solution(s)? What would you recommend today?

Tilmann Zäschke: Using an ODBMS was the right choice. It met all the requirements and made it easy to persist the complex data structure (250 persistent classes). The continuous schema changes caused by the agile development process were easy to deal with (we bundled the schema updates into batches that were released every 6-8 weeks, totaling several hundred individual changes).

Q6. Do you believe that Object Database systems are a suitable solution to the "object persistence" problem? If yes why? If not, why?

Tilmann Zäschke: Yes, in most cases. I believe small projects can benefit from the ease of use of small ODBMS like db4o. Larger projects can benefit from products like Versant ODBMS, ObjectStore, Objectivity or others. ODBMS can especially excel when persisting complex data structures, because they do not need any mapping layer (like O/R mapping). Besides being good at complex models, I think ODBMS tend to be very flexible all-rounders, which is especially interesting in agile projects, where

the persistence solution is often chosen before the requirements and access patterns are fully clear. Agile projects can also benefit from the straight forward schema evolution mechanism.
Regarding RDBMSs, I believe their strength lies especially in projects with large datasets, simple data structures and suitable access patterns. Besides ODBMS and RDBMS, there are many interesting persistence solutions that are specifically tailored to a variety of problems and which are probably hard to beat in their particular field.

Q7. What would you wish as new research/development in the Area of Object Persistence in the next 12-24 months?

Tilmann Zäschke: In research, one area where I can see opportunity for improvement include improved support database grids, also with respect to schema evolution. Another area relates to improving support for the effects of agile practices, like frequent schema evolution.
In development, I would be interested to see more widespread support for Mac OS by major vendors. I am also looking forward to the schema evolution API in the JDO 3.0 draft.