

Effective Development of High-Performance Data Services

Comparing Persistency Layers: The Best Development Procedures and Tools

The project “MINT – Model-driven Integration of Information Systems” supported by the Bundesministerium für Bildung und Forschung (Federal Ministry of Education and Research) investigated concepts and tools for the model-driven coupling between modern, object-oriented modelled business logic and relational database systems.

MODEL-DRIVEN INTEGRATION OF INFORMATION SYSTEMS THE PROJECT “MINT“

How can software systems be tailored to changing business processes and new requirements in a cost-effective way?

To answer this question, development procedures for the model-driven coupling between modern, object-oriented modelled business logic and existing relational database systems have been investigated and rated as part of the project “MINT – Model-driven Integration of Information Systems” supported by the Bundesministerium für Bildung und Forschung (Federal Ministry of Education and Research).

Here, we present selected results.

ARCHITECTURE OF THE TEST ENVIRONMENT

All applications use the same generalized test environment architecture. They focus on the different usage scenarios in order to be able to compare the competing persistence systems. Only the persistence layer that contains the specific persistence technology and the specific persistence adapter changes. The other three layers remain unchanged for the application. To receive comparable features, every technology must provide the same functionality.

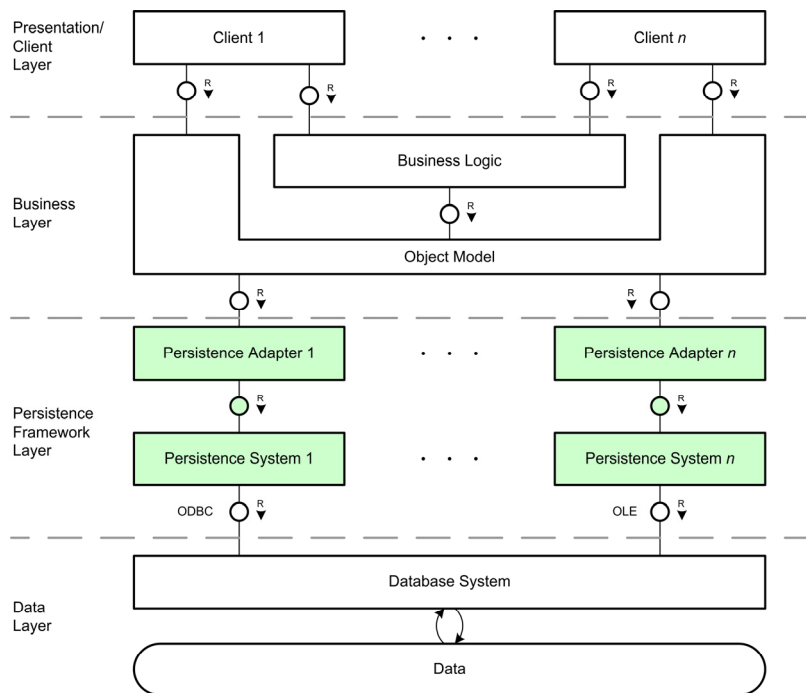


Fig: Architecture of the test environment

ADAPTER CONCEPTS

The connection, especially the object-oriented mapping, has been realized by different adapter concepts.

- Generic adapters support all test applications, but they are specific to the used DBMS as well as to the DC scheme.
They are data-centred and implement an object model according to the DB schema.
- Static solutions that have been developed for the specific object model.
- Dynamic solutions performing a runtime mapping and being configured accordingly.
- Generated adapters implementing, in contrast to the above, a mapping that is specific to the respective application.
Thus, they are application-oriented.

Test applications for the different data access variants enabled a meaningful comparison of the very different adapter concepts as well as statements about strengths and weaknesses of the individual solutions.

BEST RESULTS FOR SCORE DATA ARCHITECTURE INTEGRATION

The adapters that were fully automatically generated with SCORE Data Architecture Integration from Delta Software Technology presented one of the investigated solution alternatives. SCORE implemented the mapping between the service interfaces and the data objects in C# tailor-made for the applications, specific to the used database section and the respective DBMS.

For the adapters generated with SCORE Data Architecture Integration, the tests yield the proof of high performance, low development costs, absolute reliability of the generated code as well as drastically reduced testing effort.

TEST SCENARIOS

For the comparative tests, C# applications from a commercial banking software for business and equity analysis have been used:

- Chart Production
 - Displaying share prices (chart diagrams)
 - Read-only access, selective and batch
- Accounting
 - Corporate data management
 - Read and write accesses, selective
 - Importing share price data

ADO.NET has been used to access the real data stored in an Oracle database.

ADAPTERS IN COMPETITION

For the comparison, the adapters have been provided with the tools of the different solution concepts:

- Manually implemented adapters with ADO.NET access technology (hereinafter referred to as “ADO”)
- Persistence frameworks with configuration of the runtime O/R mapping (generic solution):
 - Telerik OpenAccess („OpenAccess“)
 - NHibernate („NHibernate“)
- Generated adapters
 - „Out-of-the-box“ code generator
 - AndroMDA
 - Modular generator framework (specifically for the development of individual model transformations)
 - openArchitectureWare („oAW“)
 - Model-driven adapter generation (tailor-made mapping for the respective application, specific to the respective database section and the used DBMS)
 - SCORE Data Architecture Integration („SCORE“)

GOAL 1: DEVELOPMENT AND MAINTENANCE

The following results have been gained during the maintenance and development of the adapters for the "Accounting" application.

| | ADO | NHibernate | OpenAccess | oAW/SCORE |
|----------------------|-----|------------|------------|------------|
| Implementation (PD*) | 4,2 | 3 | 3,25 | 0,9 |
| Test und Debug (PD) | 8,1 | 3,2 | 4,2 | 1,5 |

* PD = Person-Days

Individual assessment:

- Out-of-the-box generator AndroMDA:
Difficulties in the mapping of the database and in the adaptation to interfaces
Conclusion: Out-of-the-box generators are only partially suitable for the O/R mapping for business logic <-> DBMS.
- Manually implemented adapters („ADO“):
Highest efforts
- Persistence frameworks NHibernate and OpenAccess:
Good results for migration and evolution scenarios
Workarounds necessary
- Generated adapters:
oAW: Good results, the usage of model-driven techniques additionally reduces implementation time
SCORE: Results are comparable with oAW

GOAL 2: PERFORMANCE

The following performance results have been gained for the “Chart Production” application:

| | ADO | NHibernate | OpenAccess | SCORE |
|---|----------------|------------|------------|-------------------|
| Initialization of the application | 5.331,3 ms | 6.211,2 ms | 6.961,4 ms | 1.462,5 ms |
| Initial memory consumption after a complete start of the application | 51,8 MB | 56,1 MB | 51,9 MB | 52,9 MB |
| Time required for a selected task | 87,2 ms | 194,0 ms | 114,8 ms | 84,1 ms |
| Throughput tasks per minute | 11,5/min | 5,15/min | 8,7/min | 11,9/min |
| Max. memory usage during the execution of the scenario | 2,2 MB | 6,5 MB | 24,0 MB | 2,3 MB |
| Total of the response times of all methods of the chart production façade | 2.162,8 ms | 3.403,9 ms | 2.559,0 ms | 2.017,2 ms |

Performance times of the complete “Accounting” scenario:

| | ADO | NHibernate | OpenAccess | SCORE |
|--|-------------|--------------|-------------|--------------------|
| Execution time of the complete scenario (complex read-write-actions) | 72.077,4 ms | 113.328,0 ms | 79.938,8 ms | 63.493,6 ms |

Result:

It is obvious that project-specific adapters (fully automatically generated: SCORE, or manually developed: ADO) clearly outmatch the O/R frameworks (OpenAccess, NHibernate).

The adapters generated with SCORE Data Architecture Integration convince with low development costs, reliability of the generated C# code, drastically reduced testing effort and high performance.

SUMMARY

The Delta solution of generating the persistence adapters tailor-made for the respective application and specific to the particular database section and the used DBMS ensures high performance and comes along with high reliability of the generated source code as well as drastically reduced testing effort. The continuous tool support decisively reduces the overhead associated with the (more precisely: each) tool introduction – and thereby the development costs.

The development procedure of SCORE Data Architecture Integration clearly scored:

- The generated classes are *DB-specific* (in this case Oracle) **and** *application-specific*
- The queries and data structures are optimized application-specific
- The ADO.NET classes are highly performant because they are generated specifically for the actually required data and access structures.

It should be emphasised that the object model as well as the test application remained unchanged when generating adapters with SCORE. In the project, the generated application adapters have been completely generated from the Composition Repository data. The generated code has been generated in an (humanely) easy-readable form.

The MINT project emphasises the suitability of SCORE for the most different environments.

Once again, the tests have confirmed:

- Whatever is manually **effective** programmable, can be **generated** as well!

Automation by generative development yields:

- Reliability: Development and maintenance effort will be reduced.
- Easy, consistent and sustainable use of expert knowledge

FURTHER INFORMATION

- <http://www.d-s-t-g.com>
- <http://www.mint-projekt.de/>

The detailed project report has been published as a book:

„MINT – Modellgetriebene Integration von Informationssystemen: Projektbericht“, Publisher Prof. Ralf Reussner, Karlsruhe, 2009

The test values have been extracted from the referred project report.

DELTA



software
technology

SCORE DATA ARCHITECTURE INTEGRATION

Data as real business services:

Fast, easy und independent of data architectures and data storage systems.

SCORE Data Architecture Integration delivers exactly what is needed – data as real business services, completely generated, clearly encapsulated access code to process the information, without any pseudo-SQL interfaces.

Working with SCORE is easy. SCORE ensures that even the most complex heterogeneous data environments can simply be used as services. Even different TPMS and languages are supported, without the service consumers having to know or care about the specifics of the respective environment:

For the service consumer there is no difference between data, functions, or a combination of both – a service is a service!

WWW.D-S-T-G.COM