



photo: Provinzial

Modeling System for Insurance Products

VP/MS at the Provinzial Nord Insurance Group

By Kay Denker

Provinzial uses some applications which draw upon insurance logic. Ideally, all applications use the same logic, which is developed once and then made available as a "black box". For the construction of these black boxes, i.e. for modeling the product models, Provinzial uses the modeling tool VP/MS.

High Service Standard at Provinzial

The Provinzial Nord Insurance Group, in Schleswig-Holstein, Mecklenburg-Western Pomerania, and Hamburg, is never far from their customers: from the islands of Sylt to Rügen and from Viöl at the North Sea to Hamburg-Harburg, the net-work extends to some 240 insurance agencies of Provinzial Insurance.

Extensive services for the customer are the focus: insurance salesmen are close to the people in their region.

From car insurance to home and apartment insurance to life insurance, Provinzial has insurance products in every field, both private and commercial. Over 2,800 employees, both internal and independent, work with over a million customers with more than 3 million contracts.

To achieve this high service standard, a high-performance IT infrastructure ensures smooth execution, both internally and externally. There are two main applications at Provinzial which are the background for encapsulated logic. They are KOMPAS (Composite

Application System) and PROVIS (Provinzial Business Information System).

KOMPAS is an inventory control system. The inventory of insurance contracts includes many different contract versions for various fields. KOMPAS allows new contractual business to be processed in the same way as existing business.

PROVIS is used as advising software; in general, it works with the current version of contracts.

New Standard for Product Models

Consistency is a Requirement

In the past, the focus at Provinzial was placed on functional computing. This resulted in two separate development paths for product models, which fulfilled their own application requirements, but lost the actual idea of unified computing logic.

Accordingly, a review of existing product models was started as a project at Provinzial. The necessity of agreeing on standards was shown very quickly, because not only were different models discovered in KOMPAS and PROVIS but also within each application group.

The standard resulting from this breaks down into two areas:

- Communication between the application and the product model via a newly created unified protocol
- Formal requirements for the modeling of a product model with respect to the ease of reevaluation of product parts, maintainability of the product model, quality assurance and application stability

The product modeling is done at Provinzial by a modeling team, who are adapting the existing product model to the new standards this year and

developing new models according to these guidelines.

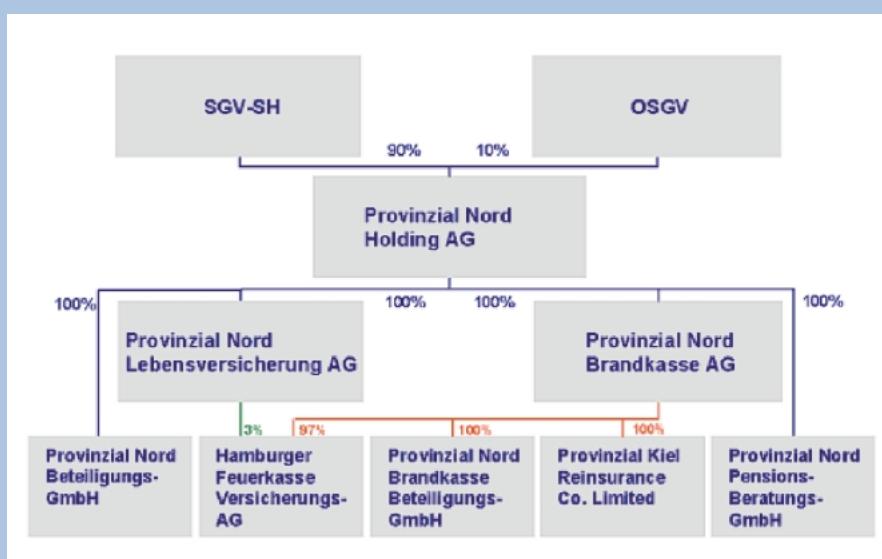
MaK DATA SYSTEM is supporting Provinzial in this project with development of the standard, coaching of the product modelers, creation of support tools, and product modeling.

What Can VP/MS Do

VP/MS is a tool for the modeling of insurance products for departments of all institutions in the insurance field. Product modeling is done using the VP/MS Workbench. The product models are provided to applications on PCs and mainframes via the VP/MS runtime environment.

Enterprise Structure

Provinzial Insurances were legally converted in 2001 from institutions governed by public law into publicly traded corporations. The sole shareholder of Provinzial Nord Brandkasse AG and Provinzial Nord Lebensversicherung AG is the newly founded Provinzial Nord Holding AG. 90% of the shares of the holding company are held by the Sparkassen- und Giroverband (Savings and Loan Association) for Schleswig-Holstein, while 10% are held by the East German Sparkassen- und Giroverband for the savings banks in Mecklenburg-Western Pomerania.



The VP/MS Workbench allows the technical definition of calculations, plausibilities, and business rules. Subsequently, things are separated into objects, events, tasks, products, components, attributes, functions, and tables. Calculations and projections are computed in forms similar to those in a spreadsheet program. The product model is testable at any point using an interpreter. A compiler then constructs an efficient runtime module from the product model.

The modeling of an insurance product with VP/MS is accurate, because a functional language is used, and complete, because it is subjected to thorough testing. Programming knowledge is, however, not required for the modeling team. A modeler doesn't need to pay attention to data types or to storage cell assignment. Uninitialized or overwritten variables, infinite loops, and pointers to incorrect regions in memory are impossible due to the functional structure

of VP/MS. VP/MS can, however, express any algorithmically solvable problem. VP/MS does its best using lazy type conversion, compilation, and extensive caching to convert the formulas given into an efficiently executable computational module.

How VP/MS Works

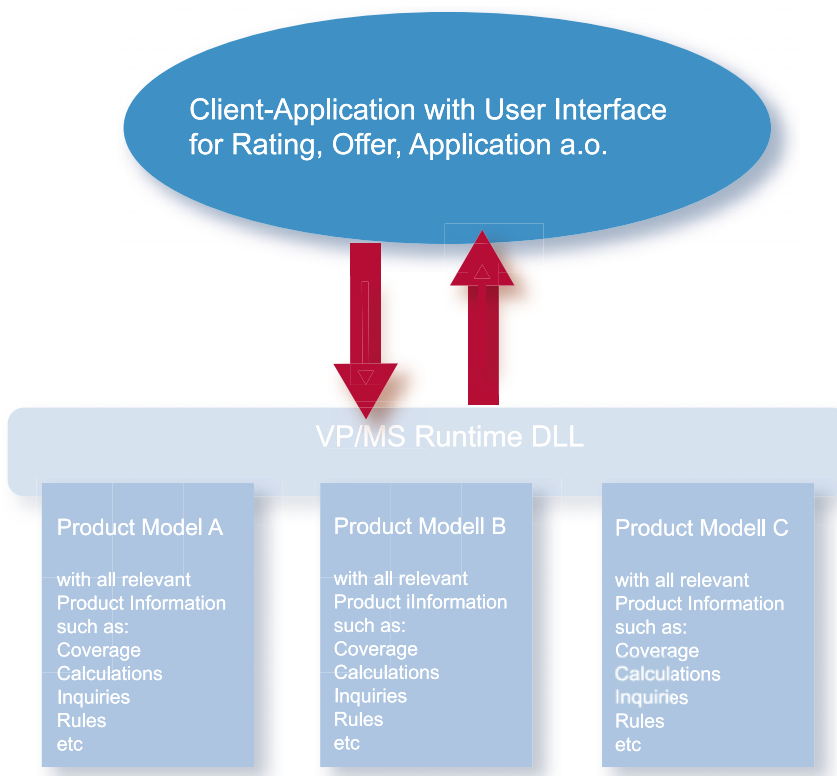
Classical programming languages are evaluated only in a single direction, that is, from input values, results are calculated. VP/MS, on the other hand, can use its computational formulas in reverse and can decide which input values are required when and for which results. The program converts its formula string into calculations, projects, and dialog form control (masking and order of fields). This automatically forces consistency in these areas.

Product models can be connected, simply to bundle contracts or to store enterprise-wide definitions centrally. Product models can invoke existing

external functions or modules from any arbitrary programming language. VP/MS's database interface allows information to be extracted from any database.

The runtime module is installed on the target machine, whether PC or mainframe, using a VP/MS runtime library. This ensures that computational results on any system will be identical. For interactive applications, the runtime environment also supplies information about the locking and masking out of input fields. Lists of values for selection fields are also provided, along with error messages and navigation between input fields.

The illustration shows the basic principles of communication using one application and three different modules. The VP/MS runtime environment and models take on the role of servers; a reaction occurs only when they are called by the application.



For further information contact:

Dr. Ingo Büll
 Phone: +49 (0)431 / 3993-554
 eMail: buell@makdata.de

Communication between an application and different models with the VP/MS runtime environment in the role of server.