Best Practices in Business Rule Design and Implementation

Moderators: Ali Arsanjani and Gilda Pour

Writing scalable, robust, and adaptable business rules for business objects in today’s e-business and n-tier architectures is a challenge. In this workshop, we categorize and capture, in pattern format, the best and successful practices relating to the design and implementation of business rules within the context of business objects and components.

Rules are the heart of business logic. Businesses implement their information systems upon the pillars of business rules. The application architecture of a business information system (whether e-business, client-server or “legacy”) handles the flow of information from layer to layer: the user-interface accepts information and filters it in a preliminary way by applying validation rules. The data is then passed through to the application layer (a.k.a. business logic layer), potentially through some middleware.

The application layer processes the data into information using the business rules that drive the logic of the application. The business application layer interacts as necessary with the data services (database or persistence) layer.

Rules in the data service layer, such as stored procedures or triggers, or plain application of rules at the data services layer through object interactions. Initially, rules were simply buried in code. Over time, as relational databases became the norm in development, rules began to be embedded in triggers and stored procedures written in various implementations of SQL.

You are trying to build systems that incorporate business rules that drive and guide business processes. The business tends to change business rules. You need traceability into the code in order to make non-intrusive changes that will have no side-effects. Nested if-then-else clauses tend to be less maintainable, scattered throughout the code and have unpredictable side-effects when modified.

You would also like to implement rules at each layer of your application architecture in a uniform, reusable and consistent manner. You would like to employ one paradigm for business rule design that is scalable throughout the layers of your application architecture. You would like to see the same sort of solution in the GUI, application and persistence (database) layers. In e-business systems, and indeed in many client-server systems, GUIs may have relatively simple validations; application layers usually contain the bulk of business processing, while the persistence layer handles database handles triggers and stored procedures in addition to simple insert/update or select operations.

The aim of this workshop is to share experiences on how to solve design and implementation problems relating to the design and implementation of business rules, constraints, invariants, policies, GUI validations, and how to cope with the rapid need to change rules as a results of changes in user requirements. Strategically, we would like to compile a handbook for the design and implementation of business rules under various project types and circumstances; what works best under what circumstances.
Some of the questions we would like to address and share experiences on are:

- What are the types of rules we most often encounter that implement a lion’s share of the business requirements?
- How should we mine business rules from within application logic which has if-else statements sprinkled within it, in order to be able to make changes to current business policies and practices as implemented by the information systems of the organization?
- How can we create adaptable and pluggable rules at various layers in the architecture: GUI, Application, Persistence, Middleware, Security, etc.?
- How do we handle the proliferation of business rules?
- How do we address scalability (number and complexity of rules), performance, maintainability and extensibility?