Functional Based Software

Steven Bucksbaum, February 5, 2011

Data Driven Software -- No Hard-Coded Customer Identifiers

Software becomes functionality based in place of customer based: Functional codes steer the program logic, not the customer identifiers. Customer identifiers and functional codes are mapped to each in database reference tables.

Development approach:

1. Identify the functionality you want to provide your customers.
2. Identify the customers who will use each functionality
3. Map the customer identifier to the functionality code in a database reference table.
4. In the program, drive the program logic based on the functionality code coming from the database reference table.
5. As new functionality is requested or discovered, add it to the database reference table as well as to the program logic.

An example: In place of coding the customer-id in the business logic, call a reference table in the database with the customer-id. The functional codes provided by the query to the reference table will provide the information of the program features the customer has access to.

Database reference tables: Customer Identifiers and associated functional codes are mapped to each other in reference tables. The functional codes are used in the program logic to allow access to program features. For example: to allow a customer access to a certain report or a group of screens, the information must first be added to the proper reference table.

Improved program flexibility: Once a specific functionality is coded and tied to identifiers in a reference table, any customer may be allowed to access that functionality. All that is needed is for the customer identifier and the functionality identifier to be linked in the proper reference table.

Understand your business domain / data model: Equally important whether you are going to design an object model or map functional activity to the customer. In some ways, thinking in terms of the functional feature set can be easier because it encourages you to think about data model separately from functions / processes but you still have to do both.

Database becomes more complex. The database is expanded to hold reference tables. The tables link the customer-id to the coded identifiers. The additional tables increase the complexity of the database.

Program code becomes more abstract: Instead of relaying on hard coded customer identifiers, the code is relying on codes provided by a call to the reference table based on the customer-id.