3 GL I/O Triggers

A Technical Manager's Approach to Extending Legacy Software

THOROUGHBRED SOFTWARE INTERNATIONAL, INC.

September 13, 2012
Authored by: Mark Lewis
Vice President Sales and Marketing

Thoroughbred® Basic™ 3GL I/O Triggers

A White Paper

Thoroughbred Basic 3GL I/O Triggers (IOT(s)) are the answer to extending legacy 3GL code and implementing modern databases with new Thoroughbred development systems. Preserve your underlying Business Rules that have weathered the test of time and that are proven to work, but enable new development with new tools and database of choice.

Thoroughbred Basic 3GL I/O Triggers can substantially reduce and simplify integration efforts to previously developed Legacy 4GL or 3GL programs, sub-systems and systems by inserting referential integrity, business rules and general processing code prior to all Basic I/O directives. In essence, tables and fields within new or old tables can be created, modified or deleted independent of the legacy program, subsystem or system. New functionality can be quickly implemented and tested without disturbing legacy code that was developed decades ago.

One of the big problems with modifying any legacy code is the destabilization of the original embedded algorithms. The utilization of IOTs to provide new system capabilities must be preceded by a design process that requires developing a block diagram of data table creation, deletion and modification. This design should start at a system high level and then zoom in on the low level sub-systems and major programs. Sub-systems and major programs must display each major database table input and output.

From a detailed analysis of the data flow through the system, the block diagrams mentioned above will help determine where the new feature IOTs must be inserted. For example: If new fields are to be added to Table-A, then each input of Table-A for the old system must have an IOT that filters the new fields so as not to confuse the old code. However, wherever Table-A is an output there must be an IOT that will insure the new fields are created when Table-A is written. If a measurement value (DISTANCE) is assumed to be meters in the legacy system, the field (DISTANCE) must have an

285 Davidson Ave, Suite 302
Somerset, NJ 08873
(732) 560-1377 • (800) 524-0430 • Fax (732) 560-1594
www.tbred.com

algorithm to convert yards to meters when the field is READ and an algorithm to convert meters to yards when the field is WRITTEN to the new database. This example assumes the current legacy system expected meters for (DISTANCE) and the new system expects yards for (DISTANCE) to be maintained in the database.

With the process mentioned above, it is evident that a new strategy of analysis, design and implementation will be employed. This approach requires that all new data must first be defined before the IOTs can be defined. This is similar to COBOL programming, and normalized database design.

What is Referential Integrity? For clarity, please find the following example: a three dimension parent child relationship: Customer Table, Order Header Table and Order Detail Table. Assume we wish to maintain a field in the Customer, Order Header and Order Detail Tables that maintains a total of all customer orders, order line item total and a line item extended total respectively. The system requirement would be: When an order line item detail changes to the degree that means a new extended price exists, the order total in the customer table should be updated and the order total in the order header table should be updated. By doing so the referential integrity of the data base is maintained. Also, writing an order item detail table with a new quantity would automatically recalculate the extended price and this would start the cascade of other updates.

What are Business Rules? Likewise, we have an example. The total line item extended total could be calculated to be (item price * quantity ordered) + shipping cost + tax. Other items like size, quantity, color, etc. could also be included.

What is a General Processing Rule? Again, we put forth an example. The item price could be maintained in several related parent child tables that require building various key values from data contained in the order and customer tables. Once the key values are built, various table(s) could be read to determine the appropriate price for the customer, examples are, quantity, discount rules, etc.

Perhaps the most significant subject could be what type of procedural/trigger code falls into the above three categories. It would be fair to define more than the three categories listed above (maybe for a

285 Davidson Ave, Suite 302
Somerset, NJ 08873
(732) 560-1377 • (800) 524-0430 • Fax (732) 560-1594
www.tbred.com

given environment there could be four, five or N categories of procedural/trigger code). When the categories are defined, the project design team is ready to start. The real truth is algorithmic procedural code is being placed in understandable compartments and can be worked on independently by project members. Further, the execution of such code during a READ or WRITE cycle can send new data to an integrated database or corporate data warehouse for further processing.

As future requirements arise, the newly created tables can have triggers attached to them so as to preserve the previous referential integrity, business rules and general processing rules code.

In the Thoroughbred environment, triggers can be attached to new database tables or legacy data files at start-up, dynamically at any execution level, or by control programming that orchestrates various system level functions. The details of the various directives and functions that support the execution, development and maintenance of low level I/O triggers can be found in Chapter 6 of the Thoroughbred Basic Developer Guide available on-line at our web site www.tbred.com.

The execution of the IOTs procedural code is done prior to the READ or WRITE to any database table or 3GL file record. By doing so, the actual algorithms of processing legacy code can be ignored and total focus and attention can be paid to the new manipulations of the data being passed from program to program, subsystem to subsystem and system to system.

As long as required, new and modified data can remain in the new database tables where standard retrieval tools can be used to query, report or process the data into meaningful information. The Thoroughbred Environment provides the assurance that the 3 GL I/O Triggers are executed as all 3GL file records and new database tables are READ or WRITTEN.

In conclusion, Thoroughbred's Basic 3 GL I/O Triggers can greatly extend the life of legacy software by enabling current technology tools, databases and development systems to be used to update the legacy system to modern standards without completely re-developing the system from the ground up, thus saving substantial time and resources.

285 Davidson Ave, Suite 302
Somerset, NJ 08873
(732) 560-1377 • (800) 524-0430 • Fax (732) 560-1594
www.tbred.com