

MultiValue Reporting and Business Intelligence

Operational Reporting versus Analytical Reporting

Ross Morrissey
International Spectrum - Jan/Feb 2006

One of the great strengths of MultiValue systems is the efficient representation of transactions; we can express complex business logic succinctly in very few files. In the right hands, our rich, expressive query environment exposes these files, extracting the detail and summary transaction information we need to run our business. But, what if we want simpler access to operational data and quick export to spreadsheets or trending and drill-down analysis to improve our business? In this article, we will look at the complementary nature of operational reporting and analytical reporting and the strengths and weaknesses of their respective applications and third party tools.

Operational Reporting

Also called Enterprise Reporting or Transactional Reporting, these are typical "canned" reports or the ad hoc queries that a MultiValue expert can pull up so adeptly in one or two commands that it seems like black magic to the uninitiated. We've all had conversations like these:

"I need a list of last month's late orders!"
"Do you want that sorted by branch number?"
"Sure - you read my mind!"

"I need a list of all customers who ordered last month!"
"Do you want contact info with those addresses?"
"Contact info too - you deserve a raise!"

Unfortunately, sometimes requests are not possible because we don't have a MultiValue expert at hand - or the information we need to run the business is not captured directly by our applications, in the form we need. Resourceful workers will gather data manually and capture it with pen or spreadsheet - anything to get their job done - unfortunately this is a time-consuming and costly approach. We get effective operational reporting only when requirements align closely with application files because the extraction process is quite limited.

The actual reporting process consists of scanning a set of records and presenting elements on a screen or page in human readable form. This approach dates back to an era when resources were very constrained. We assemble and sort a set of candidate record keys; then, in one pass, read and format or summarize each record on screen or to a spool file. This minimizes system requirements because we only hold one source record (and any running totals and page layout information) in memory during the output phase.

In spite of the single-pass mechanics, the LIST and SORT verbs, combined with the information stored in dictionaries and a variety of display options, provide an enormously flexible method of defining reports. With enough experience, it is possible to create impressive output using built-in command-line reporting. In the eighties, I attended a seminar where an elaborate LIST statement was actually used to cut automatically numbered payroll checks. Unfortunately, this is still state-of-the-art for native MultiValue

report output, and molding that output to fit the online world or populate a spreadsheet is not easy.

Ignoring output issues, a perfect report will still be out of date soon after we run it because of the nature of reports - they capture a snapshot of transaction data. Less obvious are changes to the requirements that drove the canned report in the first place. Solving these problems can turn into a real management issue. How do you rerun a report to capture the most recent data? Where is the report stored? What do we call it? We can certainly create Paragraphs or PROCs, but how are they stored, accessed, organized or maintained, and by whom? We can overwrite queries or forward queries to unauthorized parties or five different people can run the same query simultaneously - bringing our operational system to its knees.

Maintenance of queries is not the only management headache. The query language complexity and complexity of the underlying file layout also increase load on the IT department. Training end users to use select lists effectively and understand the query syntax takes time and there are few resources for this type of training. Even with report training, exposure to the application dictionaries can often overwhelm the keenest end-user. The net result: IT staff will be generating reports as long as they admit to knowledge of that black art, and they will always be behind the curve.

Guided Operational Reporting with Report Generators

One solution to simplifying complex tasks is a "wizard" approach - breaking the reporting task down into smaller steps guided by a third party report generator. These have been around for a number of years and have recently embraced the familiar web interface. Training is minimized because report generators hide much of the complexity of the underlying query language, and it is getting possible to create a report without direct knowledge of the underlying file structure. The earliest Report Generators checked query language syntax and prevented the creation of invalid queries. Today, Report Generators are also aware of file layouts and can help create queries that are both efficient and more likely to create desired results. Additional features include user-friendly menus and a security layer that controls

Other approaches to Operational Reporting

We can also use ODBC or XML to pull operational data out of the MultiValue environment in order to facilitate reporting. In the September/October 2002 International Spectrum Magazine, author Michael Ballard devotes most of MV Report card: Lessons Learned In MultiValue Reporting to the overhead associated with ODBC, primarily to enable mainstream reporting tools like Crystal Reports to access MultiValue data. Michael does not paint a pretty picture. More recently, new middleware based on Microsoft's .NET technology has emerged as a possible reporting solution. The fact that .NET is based on XML solves some of the structural issues with ODBC, but it is an expensive proposition if your organization is not leveraging the .NET middleware for operational use too. Another solution using XML to preserve the MultiValue structure of the exported data is the new MITS Report product, described in the sidebar.

Limitations of Operational Reports

Let's revisit those operational queries that we so easily dispatched earlier, but with an added wrinkle that will sound both annoyingly reasonable - and impossible with our "single-pass" query tools:

"I need a list of last month's late orders!"
 "Do you want that sorted by branch number?"
 "Sure - can you give me the percentage change from last month in order count and dollar volume too?"

"I need a list of all customers who ordered last month!"
 "Do you want contact info with those addresses?"
 "No, but I only really need the top 10 customers last month"

These kinds of queries are typical of analytical reporting. They are essential in today's business climate as managers seek to identify under- and over- performing aspects of their operations to reduce costs and improve results. They also underline the two main limitations of traditional reporting against operational files - the restriction to sorting by value forced by the single-pass technology and the inability to compare multiple sets of data (figure 1).

Figure 1:

Operational Reporting

Report	Incident	Date	Victims	Type	Offense
2003-031125	2003-0188492	05/31/03	1	0625	RESIDENTIAL BURGLARY
2003-018208	2003-0111795	06/15/03	1	0625	RESIDENTIAL BURGLARY
2003-017166	2003-0105325	06/29/03	1	0625	RESIDENTIAL BURGLARY
2003-016011	2003-0097870	06/30/03	1	0610	BURGLARY
2003-016000	2003-0097767	06/30/03	1	0610	BURGLARY
2003-015997	2003-0097734	06/30/03	1	0610	BURGLARY

Data are sorted by ID
 Rows may be sorted or summarized by a data value, but cannot be sorted by aggregated totals

Single Data Set Rows
 Each row contains information about the same item or summarized items

This progression from operational to analytical reporting mirrors the role of the report recipient. An operational report might supply a list of late orders to help operations improve their turnaround. Middle management might look to a tactical report identifying branches with above or below average turnaround trends to gain insight or focus attention, while upper management may look at enterprise-wide turnaround figures to see if they meet strategic targets. An exciting prospect for analytical reporting is the ability to supply these advanced tools to people in the field, allowing them to act as change agents pursuing goals instead of waiting for guidance.

If you are not providing this analytical capability through application programs or online analytical processing (OLAP), resourceful managers are surely taking pencil or spreadsheet in hand to get the job done on their own. For a sole proprietorship that may be fine, but the astounding return on investment for OLAP tools means almost any size organization can benefit.

Since operational reports work on one pass through the data, we cannot easily use them to compare different sets of data, for example comparing sales numbers from month to month or ranking by summarized totals. It is possible to accomplish this analysis through custom programming, and some vertical applications do this with or complex multi-pass BASIC code; typically generating a summary history file with delimited keys of the form

year*period*product-type*product. This programming approach can yield powerful results that would not otherwise be available, but there are a few drawbacks.

Generating management reporting with analytical capabilities using BASIC programs involves an intimate knowledge of the underlying application as well as detailed insight into the reporting requirements. The authors of the application software may have the former but lack detailed specifics of the latter. Putting the hooks into your application to extract transactional information and load it into a summary file adds complexity to your code base. If you bundle this with transaction processing, you slow operations down; otherwise, you create another process to schedule and administer.

Reporting from these summary files can be performed with command-line queries, report generators, or programmatically. If you use traditional reporting methods to access this data, you are subject to their presentation limitations and interactive analysis is not possible. It is possible to create links manually between queries in some modern report generators, emulating interactive analysis, but mainly you gain the ability to view results easily online or to export to a spreadsheet for further manipulation. If you use a BASIC program to access the data, you can tailor the reporting, at the expense of even more added complexity - but users may still press for spreadsheet integration.

Business Intelligence and Analytical Reporting

A more general approach is using third party business intelligence OLAP or analytical reporting tools. These copy the transaction data into a structure optimized for comparing and exploring summary totals with linked and pre-aggregated sets of data to enable trend and drill-down analysis. This creates a view of a process that allows you to derive insight quickly to help you improve that process (figures 2-4).

Figure 2:

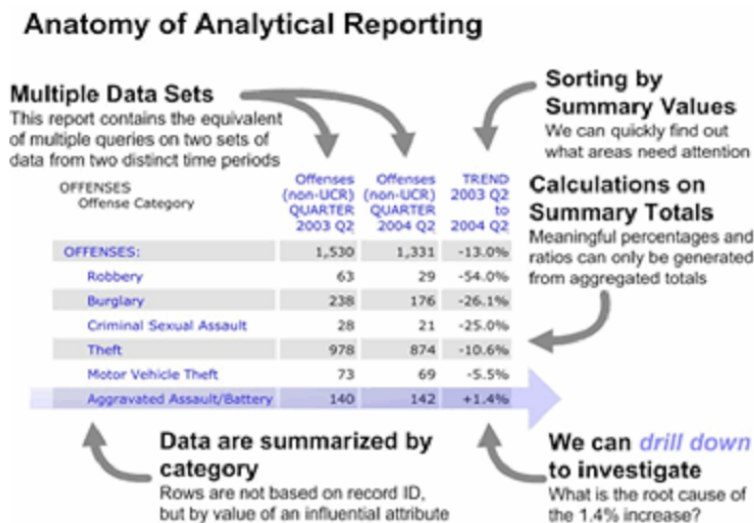


Figure 3:

OFFENSES Offense Category Area	Offenses (non-UCR) QUARTER 2003 Q2	Offenses (non-UCR) QUARTER 2004 Q2	TREND 2003 Q2 to 2004 Q2
OFFENSES:	1,530	1,331	-13.0%
Aggravated Assault/Battery	140	142	+1.4%
Area 1	67	50	-25.4%
Area 2	57	79	+38.6%
Area 3	15	11	-26.7%

Root Cause Investigation

Drilling down in to the Aggravated Assault category shows that only Area 2 is on the increase – we can **drill down** further...

Figure 4:

OFFENSES Offense Category Area	Offenses (non-UCR) QUARTER 2003 Q2	Offenses (non-UCR) QUARTER 2004 Q2	TREND 2003 Q2 to 2004 Q2
OFFENSES:	1,530	1,331	-13.0%
Aggravated Assault/Battery	140	142	+1.4%
Area 2	57	79	+38.6%
District 2	8	22	+175.0%
District 4	5	22	+340.0%
District 6	25	24	-4.0%
District 8	19	11	-42.1%

Root Cause Investigation

We quickly isolate the city-wide increase in assault to Districts 2 and 4 within Area 2

Although business intelligence (BI) draws information from the underlying operational data, it is actually modeling the business process, not just assembling statistics. Reporting directly from operational data files to operate a business from day to day does not constitute BI. Reporting with a goal to improve the process, to track trends, to identify under-performing categories and the underlying root causes are the hallmarks of BI.

Operational systems break down a business process into individual transactions that can be performed independently, reliably, and efficiently; BI systems look at the business process as a whole and involve trade-offs that would be impractical in transactional systems. To provide rapid on-line analytical processing, we pre-aggregate transactions in a structure optimized for efficient navigation. Since each transaction can affect hundreds of elements in this structure, it is impractical to update this structure during the actual transaction.

Most BI solutions involve batched extraction and summary of transactions into a

hypercube structure. We can usually make these batches arbitrarily small to allow transactions to flow into the hypercube in near real time without putting operational transaction response time at risk. In OLAP tools other than MITS Discover, any update requires exporting the batches of transactions out of the MultiValue environment into a proprietary OLAP tool or staging area, adding complexity to the solution and limiting operational integration with the OLAP tool. MITS Discover runs within the MultiValue environment which greatly reduces the complexity of using this solution.

Using Hypercubes for OLAP Storage and Navigation

The standard approach for analytical reporting is the OLAP hypercube. Hypercubes arrange aggregate transaction totals along dimensions like customer, product, and date. We can obtain the transaction totals for any combination of dimension values by directly reading data from the hypercube instead of scanning all the source data. The great strength of the hypercube is that the number of underlying source transactions does not affect query speed. The hypercube also facilitates slicing and dicing - swift navigation from summary totals to finer-grained subtotals.

Examples of this technique: slicing subtotals for last month's most profitable sales rep to discover their product line mix and how it differs from under-performers, or discovering how community policing efforts have reduced property crime by area to evaluate program effectiveness.

The advantages to using an analytical reporting tool over writing BASIC programs for management reporting are similar to the advantages report generators have over the command line. Certainly, knowledge of the application and reporting environment is required, but we view these through the lens of a BI tool, not directly using BASIC programming and manual file access, which dramatically reduces the effort and complexity of the solution. We manage and schedule extraction and loading of transactional data automatically. The biggest benefit is the intuitive interaction of the GUI OLAP clients available for analytical reporting tools with their ease of export to spreadsheets.

Summary

This progression from manual approach through application development to managed third-party solution is similar for both operational reporting and analytical reporting. It may benefit your organization to move toward the managed end of the spectrum with its increased productivity and capability. There are several MultiValue reporting solutions available. Organizations will benefit most by identifying what their separate needs are for operational reporting and analytical reporting, and then finding the solutions that best satisfy both needs.