Changes of Degree
vs.
Changes of Kind

INTERNATIONAL Spectrum®

2013 MultiValue Conference

Why Attend?

It is difficult for technology managers and senior data management technicians to find meaningful training that can provide both breadth and depth in targeted technologies.

The International Spectrum conferences have proven, for many MultiValue professionals, to deliver the best value as a training alternative to pure depth training (e.g. a 4-day class on Database Performance Tuning). The quality and quantity of educational value of the International Spectrum conference has been consistent and cost beneficial for over 29 years.

The conference offers nearly one hundred technical sessions. The presentations are conducted by a mix of MultiValue users, developers, tools providers, database providers and consultants. Throughout the conference, there are ongoing opportunities to discuss technology with MultiValue developers, fellow users, and vendors across a broad continuum of database, tools, and related product issues.

Who Attends This Conference?

Executives, Chief Information Officer (CIO) and Executive and Senior Level Management, Owners, Consultants, Influencers, Technical Staff, Implementers, Sales, Operations and Production, Industry Leaders, and Vendors.

Presenting to Management:

For resources to help you present to your management why you should attend, go to the following url: http://www.intl-spectrum.com/conference/WhyAttend.aspx

Register Before March 6th and Save $300!

INTERNATIONAL SPECTRUM 2013 PACKAGES	Price On/After 03/06/13	Price Before 03/06/13
❏ **Full Conference Package**	**$1395**	**$1095**
❏ **One Day Conference Package**	**$895**	**$895**
❏ **Group/Team Package** 3-5 named attendees from same company	**$2985**	**$2985**
❏ **Professional Networking**	**$75**	**$75**
❏ **Exhibits Entrance Pass**	**$35**	**Free**

April 15 – 18, 2013 | The Wigwam | Litchfield Park, Arizona | www.intl-spectrum.com/conference/2013

INTERNATIONAL Spectrum

THE MULTIVALUE TECHNOLOGY MAGAZINE

6 **Business Tech: Degree and Kind** In our Information Management industry, we are used to change. With our MultiValue databases and tools, we sometimes even revel in change since it is another opportunity to show how quickly we can do things. But in our praise of change and rush to coding, we sometimes forget to think about the change, even starting with analyzing its basic nature. **BY CHARLES BAROUCH**

FEATURES | JANUARY/FEBRUARY 2013

8 **Locking – Part 3: Modern Locking** The first two articles of this series examined the issues of locking as it applies to database integrity, information correctness, and the traditional manner in which most of our MultiValue applications implement locking. This final article now moves on to explore the locking requirements of modern web applications, web services, and other forms of SOA. **BY BRIAN LEACH**

16 **The (MultiValue) Dating Game** We all know how easy it is to work with dates in MultiValue databases and programs. This article takes it several steps further than the simple "end date minus start date" calculations and also provides code example that may find a niche in your utility BP library. **BY KEVIN KING**

From the Inside

Do you think you are a good coder? Have you been practicing your coding, or have you just been doing your job?

If you haven't been practicing your coding, then you are likely getting rusty in it. When you are a musician or a sports star, you have talent and practical demonstrations of what you can do, but you practice, practice, practice.

Most of us assume that our day to day software development is our practice. But this is not the case. Let's use basketball as our example. Your day to day software development is much like an NBA Game. This is where players make their money, but they spend most of their time reacting to the game and other players.

Players then spend time between games to practice their jump shots, layups, and passing in order to do better in the games. This is what you as developers need to do.

This is also where "Code Kata's" come into play. Code Kata was a term coined by Dave Thomas, co-author of the book The Pragmatic Programmer. A Code Kata is an exercise in programming which helps a programmer hone their skill.

I thought this was a novel concept, and decided it would be interesting to introduce the concept with Multi-Value Developers.

Let's look at Dave's First Code Kata (pricing goods at supermarkets) (intl-spectrum/s1058). Kata One – Supermarket Pricing

Some things in supermarkets have simple prices: this can of beans costs $0.65. Other things have more complex prices. For example:

- Three for a dollar (so what's the price if I buy 4, or 5?)
- $1.99/pound (so what does 4 ounces cost?)
- Buy two, get one free (so does the third item have a price?)

This kata involves no coding. The exercise is to experiment with various models for representing money and prices that are flexible enough to deal with these (and other) pricing schemes, and at the same time are generally usable (at the checkout, for stock management, order entry, and so on). Spend time considering issues such as:

- Does fractional money exist?
- When (if ever) does rounding take place?
- How do you keep an audit trail of pricing decisions (and do you need to)?
- Are costs and prices the same class of thing?
- If a shelf of 100 cans is priced using "buy two, get one free", how do you value the stock?

-**NATHAN RECTOR**
President, International Spectrum
nathan@intl-spectrum.com

twitter.com/intlspectrum   intl-spectrum.com/facebook

Degree and Kind

BY CHARLES BAROUCH

Programming and Analysis are rule-based behaviors. When you program, you are teaching the computer to apply a specific set of rules. When you do analysis, you are testing data against a pre-defined set of rules. When the degree (scale) and kind (approach) change, we have to watch to make sure that our assumptions change as well.

A Kind Thought

When talking to a former U2UG board member one afternoon, in the middle of my explaining a problem he asked me: *"Is it a change in degree or in kind?"* While I had often thought in kinda/sorta those terms, I had never heard it reduced so effectively to one question. Sometimes the shifts in business are degree — going from twelve locations to thirteen — and sometimes they are more severe — going from direct sales to channel sales — which are changes in kind.

While you might think degree changes are easier, they are not always a simple as we'd like to think. As for kind changes, let's just say they are unkind.

Ware Are You?

Imagine that your company has always warehoused goods at your headquarters because that's the only location you have. Expansion allows you to open a second

While you might think degree changes are easier, they are not always a simple as we'd like to think. As for kind changes, let's just say they are unkind.

location which will, in this example, take over 100% of the warehousing. This is a small shift which might require some re-tooling, but not nearly as problematic as it will be when you say "take over 99%" instead of 100%. Why is that 1% so difficult?

In the original plan, Bin A was always walking distance from Bin B. With one and only one location, you never have a travel barrier to combining items into one package for fulfillment of an order. With one location, you have the option of having a rule which says that all goods of the same type must reside in the same zone. This would certainly make it easier for picking and packing. It would make it easier for inventory control. However, if you have two locations, any given item has two zones, minimum: a zone in the HQ and a zone in the warehouse.

Completing an order for shipment might now require a picker/packer who can drive between bins as walking is no longer possible in many cases. Since we don't want drivers dispatched for each order, we have a planning step that was unneeded in the old (100%) way but needful in the 99% (or lower) case.

Worse, what if the HQ has a special environment, like a freezer vault or a secure room, but the warehouse doesn't have one. Now, most goods have two zones: one at HQ and one at the warehouse. Some goods can only have one zone: HQ.

1 + 1 = 3

As if that were not bad enough, adding a second location automatically requires a third location: in-transit. Imagine doing inventory while goods are being moved to HQ — samples for example — and you count each location: one hundred in the warehouse and zero in HQ. If there are fifteen on the truck, you just under counted by fifteen *unless* you treat the truck(s) as a third location.

If we just think in terms of locations, "one plus one equals three" covers most of our issues. However, we have to look beyond that to the business processes. Let's make the huge assumption that your business has a website and that the site has pictures on it. In the original scope, you had one location: HQ. If you wanted a picture of a product in inventory, you walked through a door from the offices to the embedded warehouse, took the item from the shelf, walked back into your office, and took a picture. Now, with 99% of the products at the warehouse, you either need to move the photographer or schedule shipments of product between locations just for photo ops.

Likewise, if you show products to clients at your headquarters, or if you hand out samples to your sales staff when they come "home" between trips, however you use products beyond simply fulfilling order: all of that needs a rethink.

Kind of Like That

For a change in kind, we have to work even harder. Imagine that your business is product-centric. The edict comes down that you are going to start adding warranty service to your sales mix. To management, service might seem to be just another SKU (Stock Keeping Unit). But to you, and the people who have to manage it, the difference between product and service is huge. Especially when you add the word "warranty" into the conversation.

A brief comparison between the three models should get us on the right track:

1. Product Model:

   Sold in SKUs.

   Restock involves manufacture, import, or purchase.

   Shipping is a factor.

2. Service:

   Sold in SKUs (blocks of time consumed).

   Normal restock is automatic: new days keep arriving for the schedule.

   Expanded restocking requires hiring.

   Scheduling needs to be done.

   Travel is not the same as Shipping.

3. Warranty Service:

   Price is not related to blocks of time consumed. Some warranties will be paid with no requests against them, some will require a great deal of time. Price does not reflect individual usage, it must be aggregated across all sales.

   Scheduling needs to be done, as frequently as needed.

   Warranties may also involve replacement parts — i.e. decreases in inventory without one-for-one increases in payments.

   Travel is still not the same as Shipping.

4. In all cases:

   Might have discounts for multiples being purchased.

   Might have tiered pricing (Distributors vs. Resellers vs. Street price).

   Products might expire, service hours always expire (you can't sell January 2013 hours in February 2013).

This is not an exhaustive list, but it is enough to give you a sense of the issues. Changes in kind are seismic shifts. They require extensive rethinking and detailed planning. Force fitting your existing model might be expedient but you will keep paying for that initial speed throughout the lifetime of your software.

What a Tangled...

A perfect example of a seismic change is adding a web store to your existing sales model. People think a website which sells is just an electronic version of your catalog. Doing things this way — thinking that web is a change in degree instead of seeing it as a change in kind — is going to lead you to a dead end. Why? Two main reasons: (A) because customers use paper differently than they use the web, and (B) they each offer features which the other lacks.

As potential customers, we can easily annotate a paper catalog, circling things of interest and using the — comparatively — infinite space of our physical desk to spread out pages torn from the catalog for comparison. On the web, I can reconfigure my — comparatively — limited screen space to see the products I want to compare side-by-side.

In a catalog, people will often enjoy reading long, rambling exhortations about a product or services. On the web, they expect shorter text with more pictures or white space to break it up. we can keep going one with point-by-point differences, but all you need to do is visit a dozen of the most successful sales sites and then look at a dozen successful catalogs. They each use

their medium to best advantage. They do not look the same.

Of course, tablets are starting to blur the line again. Holding a tablet they way I hold I book makes me read sites more like I read books. It sounds stupid, but it is becoming a common experience. You can find people all over that would find reading a book on the computer screen a chore but easily read on a Nook or Kindle.

The Nth Degree

Qualifying a project as degree or kind helps us determine the type of analysis we need to do before making changes. Just knowing how different these two classes of work are helps us manage expectations — those of others and even our own. **IS**
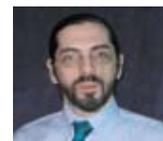
CHARLES BAROUCH is the CTO of HDWP, Inc. He can be contacted at www.hdwp.com

LOCKING

Part 3: Modern Locking

BY BRIAN LEACH

Breaking the Model

In the previous article we looked at the traditional MultiValue locking model and also mentioned the three approaches to locking: pessimistic, optimistic, and merging. We also covered some of the traditional problems associated with enforcement, lock promotion and user feedback.

MultiValue developers have wrestled with these for a long time and their impact is generally understood: the widespread use of 4GLs and runtime clients has been a helpful factor in this regard as they generally offer a consistent approach to locking. It is when application move away from the traditional single-screen logins to the desktop and the web that the traditional locking model really struggles.

Persistence and Sessions

The locking model was designed at a time when, generally, entry screens did one thing at a time. The Basic language is effectively single threaded, and you can only run one program at a time. Users would start processing, do stuff, and finish a process before moving on to the next thing. Under this architecture, it was good enough for locks to be identified as owned by a specific user or session.

The cracks started to appear even with text based systems like SB+. This went some way to emulating a persistent forms application by allowing screen stacking, in which users could quickly open a new screen by entering shortcuts in any field, making for a very flexible experience (fig. 1).

Unfortunately, it was also possible to jump back into the same screen at a lower level and to load the same record into that screen with no complaints from the lock system. As far as the database was concerned, you already own that lock, so you have the rights to that record. Suddenly you ran the risk of overwriting and losing your own updates!

Now consider what happens in a modern, GUI-based system. Along with the many changes this introduces, possibly the biggest is the shift is to an architecture in which multiple forms can persist concurrently, all connecting through to the same session (fig. 2).

This type of architecture introduces three new problems.

First, the number of records visible is likely to increase: users are more likely to keep a number of screens open, increasing the number of locks held and potentially the length of time a lock is held. That in turn means a greater impact on the lock table and lock management.

Second, the locks are still owned by the session. As with the old SB+, you can open the same record independently in multiple forms without the locking system intervening at the database. This leads to potential loss or overwriting of updates unless your GUI explicitly blocks or warns against this in its own code by maintaining its own internal lock list or checking all open forms when a record is loaded.


Id: 123 → Customer (123)

Id: 28 → Order Entry (28)

Id: 123 → Customer (123)
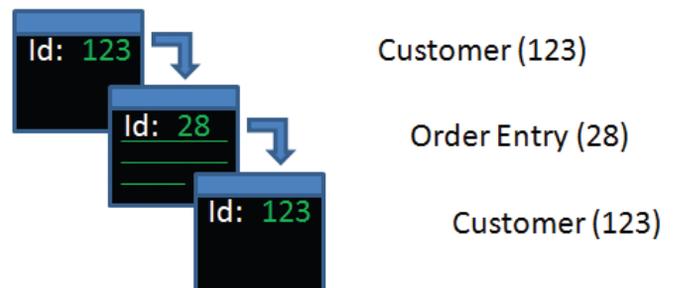
Fig. 1 Screen Stacking in SB+

The third problem is one of MultiValue database design. Traditional MultiValue applications traditionally link parent and child records using lists of multivalued foreign keys on the parent record. This differs from mainstream design where the parent id is held on the child and indexed to join the two together. Partly this is the result of historic accident — the lack of good indexing features on the older databases — and partly the nature of the enquiry language that uses file translations to pick up the details from related records, so using multivalued keys to look up the details of the child record from the perspective of the parent.

Consider an agile task management application like the one above. Here, a backlog of user stories are segregated into tasks and organized into sprints that in turn belong to releases. There are several hierarchies here, including:

```
Release -> Stories -> Tasks

Release -> Sprints -> Tasks
```

In an old-style design, the release record would contain a multivalued list of keys to related stories and sprints, both of which in turn would contain multivalued keys to tasks. But for a GUI this becomes more dangerous. What if the child and parent are both open in different forms as part of the same session? If you call a server side subroutine to add a new sprint, how does the copy of the release record held in the GUI know that the copy on the database has changed? The lock system shows them owned by the same session, so it will not intervene to tell you.

The solution is, of course, to separate out the index from the parent: adding a secondary index to the child record holding the parent key, as you would on a relational database (here being the release key on the sprint and the story) or breaking out the lists from the parent record and holding them in another file that is never
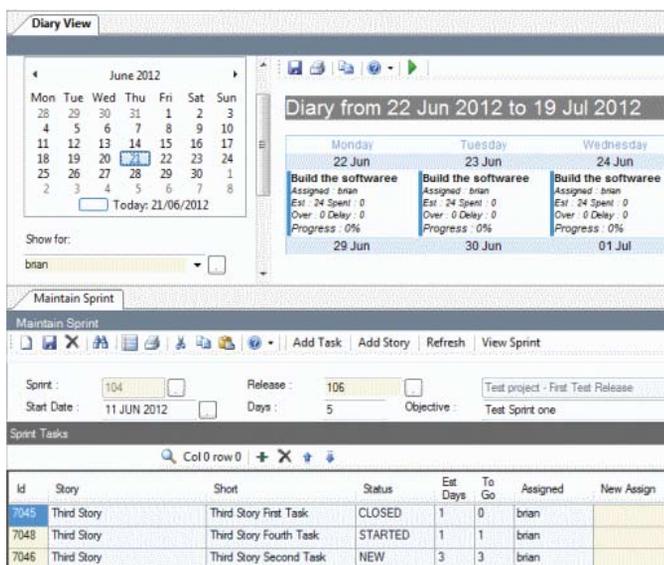
Fig. 2 Modern GUIs present multiple concurrent forms.

directly loaded into the GUI, thus retaining consistency with the enquiry language that now has to do two translates instead of one:

```
Release -> Release/Sprint index -> Sprint ->
    → Sprint/Task index -> Task

Release -> Release/Story index -> Story ->
    → Story/Task index -> Task
```

This keeps it all nice and neat, but introduces changes to the database design and involves more work in maintaining these — unless you want to move to a more restricted, primitive GUI in which the visibility of records is more limited and more in line with the older text based applications.

This, ironically, is often the case with web applications — though here the problems with the locking model are more obvious and sometimes more difficult to solve.

Stateless Applications

Web applications, web services, and other forms of SOA are designed to be stateless. Each and every call to the database exists in splendid isolation and it is not guaranteed that successive calls made from the same client will be handled by the same server process. In other words, the link between a user running an application and a database session goes out of the window.

Where the locking model is based on the concept of a user requesting an intent lock before performing any updates, and those intent locks are identified by the database session, this clearly is not going to work for a stateless application. Not only will the session requesting a lock be different than the session committing any eventual update, there is the question of lifetime.

In the traditional locking model, you can retain a lock for as long as you know a user is interested in updating a record, and release it when they are finished. In a stateless application you never know when the user has finished — they may start an update, give up, and close a browser window and you will be none the wiser. Unlike a terminal or client/server session, you never get a message to tell you the user has logged off and gone home. And writing soft locks to control access isn't a reliable option, even with timeouts.

For stateless applications, the only real options are to use optimistic or merge locking.

Optimistic Locking

Optimistic locking means allowing a user to perform updates against a local copy of a record in the hope that the underlying record has not changed in the interim. Merge locking means an expectation that the record may have changed, and taking appropriate steps to resolve the conflicts.

LOCKING – PART 3: MODERN LOCKING

To perform either of these against a single record you need three things:

- An original copy of the record as it was when you read it from the database.

- The current copy of the record as held in the database for comparison.

- The current copy of your record with any changes to apply.

- From a processing point of view you need:

- A way to quickly and effectively identify whether intervening changes have taken place.

- A way to lock against other changes whilst applying any conflict resolution.

- A strategy for rolling back an update or for conflict resolution.

Locking in the Wider World

I began this series by quoting a U2UG post complaining that newer MultiValue developers don't understand locking. So it may not surprise readers that a frequent complaint amongst mature SQL Server developers (and that is almost as old as PICK!) is that younger programmers don't understand locking either. But to understand optimistic locking it is a helpful diversion.

If you have a copy of SQL Server or SQL Server Express and a copy of Visual Studio or Visual Studio Express, try the following:

First, create a database table with four columns as follows:

```
CREATE TABLE [dbo].[locktest](
    [first] [int] NOT NULL,
    [second] [nvarchar](50) NULL,
    [third] [nvarchar](50) NULL,
    [fourth] [text] NULL,
CONSTRAINT [PK_locktest] PRIMARY KEY CLUSTERED
(
    [first] ASC
)
```

Second, create a new Windows Forms application and from the Data menu select New Data Source -> DataSet. When it asks you to select your tables for the DataSet, choose the locktest table above. Visual Studio will create a DataSet including the locktest table and generate Fill and GetData methods to populate it.

Now you're going to get Visual Studio to create an update method for you, such as you might do as a junior SQL programmer. Double click the DataSet in the Solution Explorer and navigate to the UpdateCommand for the TableAdapter at the bottom (fig. 3).

Double click to see the query that Visual Studio has generated:

```
UPDATE locktest

SET first = @first, second = @second, third
    → = @third, fourth = @fourth

WHERE (first = @Original_first) AND (@IsNull_second
```

Fig. 3 Dataset with Update Command

```
= 1) AND (second IS NULL) AND (@IsNull_third = 1)
AND (third IS NULL) OR (first = @Original_first)
AND (second = @Original_second) AND (@IsNull_third
= 1) AND (third IS NULL) OR (first = @Original_
first) AND (@IsNull_second = 1) AND (second IS
NULL) AND (third = @Original_third) OR (first = @
Original_first) AND (second = @Original_second) AND
(third = @Original_third)
```

Ouch! All that just for one simple four column table — and as the table becomes more complex so the command becomes longer and more inefficient. What's more, this is only recognizing whether the underlying record has changed in the meantime, and will throw an error if it has.

Fortunately for SQL Server developers, you can add a new column to the table as follows:

```
DROP TABLE [dbo].[locktest]
GO
```

```
CREATE TABLE [dbo].[locktest](
    [first] [int] NOT NULL,
    [second] [nvarchar](50) NULL,
    [third] [nvarchar](50) NULL,
    [fourth] [text] NULL,
    [sequence] [timestamp] NOT NULL,
 CONSTRAINT [PK_locktest] PRIMARY KEY CLUSTERED

(
    [first] ASC
)
```

If you now repeat the steps above you will see a much shorter statement:

```
UPDATE    locktest

SET       first = @first, second = @second, third
    →  = @third, fourth = @fourth

WHERE    (first = @Original_first) AND (sequence
    →  = @Original_sequence)
```

The timestamp column automatically stamps any changes made to the table allowing Visual Studio to recognize that an intervening update has taken place. Timestamp incidentally has no relation to date or time, but is just a measure of serialization. You can easily do the same with your own MultiValue database files by including an update stamp field for your own conflict checking though you must populate it yourself. Unfortunately, UniVerse users have no way to tell Visual Studio that a field is a timestamp column so it will continue to generate the same verbosity if you use any of the relational APIs.
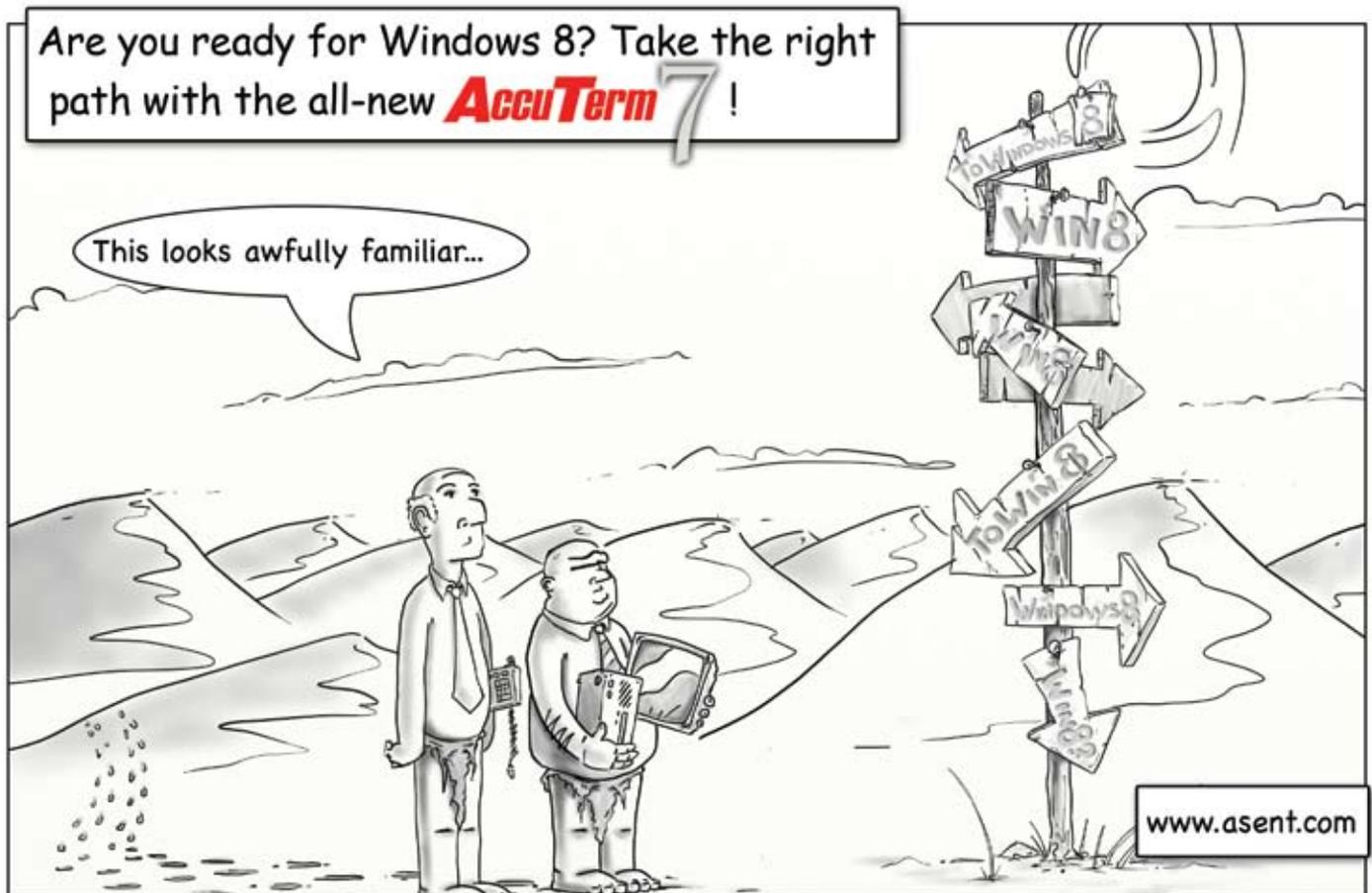
Rocket Software's CorVu NG business intelligence product now works with its UniData and UniVerse databases.

Rocket Software, Inc.(www.rocketsoftware.com) has announced that CorVuNG is now fully compatible with Rocket Software's U2 products — the UniData and UniVerse databases. U2 customers who use U2 DataVu will be transitioned to CorVu NG at no cost. With CorVu NG, users of the U2 databases will now have access to many additional features and benefits of the Rocket CorVu product suite, including drill-down path support for multiple paths or queries, allowing organizations to explore data more easily, and Aerotext text analytics, which will now be automatically available for customers who switch to CorVu NG.

Peter Richardson, Vice President and General Manager of Rocket's business intelligence and analytics business unit, says, "One of the biggest advantages that our valued customers gets making this change is getting access to a wider range of Rocket services and products to help them reach their business goals. CorVu NG has the ability to work with the full suite of CorVu performance management modules, including CorStrategy, CorPlanning, CorRisk, and CorProject. This is a major enhancement for U2 customers who are interested in elevating their BI solution to one that includes not only tactical performance visualizations, but also strategic, high-level KPI tracking."

"The addition of products from Rocket Software's other brands to our existing U2 distribution portfolio provides our partners with new opportunities to expand their market," says Brian Egan, U2 Channel Manager at Meier Business Systems, an Australia-based distributor for Rocket Software. "By adopting the 'One Rocket' philosophy, our customers can now be better positioned to deliver a broader range of solutions which not only complement but also supplement their own solutions."

In addition to introducing CorVu NG for U2 users, Rocket has also added key content to the new U2 web site (www.u2.rocketsoftware.com) including data on SB/XA, information about the new CorVu NG software, and a new "professional services" section. The site also includes Rocket U2 Technology Solutions, and the "Partners Solutions" section has been subdivided into "Solutions by Industry" and "Solutions by Business Need" to make it easier for partners and users to find the relevant information they need.

About Rocket Software

Rocket Software (www.rocketsoftware.com) is a global software development firm that builds enterprise products and delivers enterprise solutions in the following segments: Business Intelligence and Analytics; Storage, Networks, and Compliance; Application Development, Integration, and Modernization; and Database Servers and Tools. Rocket is engaged in business and technology partnerships with IBM, EMC, Fujitsu, HP Enterprise Services, Hitachi Data Systems, Avaya, Motorola, Epicor, and many others. The company is headquartered in Waltham, Massachusetts, USA. ∎

UnForm Accounts Payable Workflow Solution From Synergetic Data Systems, Inc.

UnForm Accounts Payable (AP) Workflow Solution is a flexible accounts payable document workflow and approval system for documents archived in UnForm archive libraries. ERP reports like Aged Payables and Payment Selection reports can query the workflow system to display document status prior to releasing invoices for payment, helping to improve internal accounting controls, manage payables, and reduce paper handling.

Features:

- Simple setup and configuration of Workflow
- Scanning capture job definitions
- Outlook Addin for email capture of documents
- Mergeable "startflow" rule-sets for integration with outside form or report rules
- Define custom linear flows for approval and other tasks
- Define and assign tasks to users
- Define task status categories for interim status prior to approval or handoff
- Tasks can share task status categories or have their own set
- Standard trigger events for moving documents between tasks
- Note-taking feature with task and user cross-reference
- Browser and PDF user task lists with links to Voucher Packet PDF with cover page and document images, status update web form, AP document property edit form, and status/notes/log report viewing
- AP library security measures to prevent non-authorized approvers from viewing or changing documents that are not in their own task list
- "Capture 2 ERP" batch processing infrastructure for XML file import to ERP AP Entry
- User signature stamp feature
- Task and user proxies for temporary task reassignments
- Action alerts – configurable notification by user
- Web-form prompt label editing for web form customization, including html tags
- Extensible document properties

To see just how the UnForm Accounts Payable Workflow Solu-

tion works contact Dan Schmitt, 800-446-7374 ext. 125 or dans@synergetic-data.com ∎

Reality V15.0 Latest Release from Northgate IS

Reality V 15.0 was released in December 2012, the latest from the providers of the first commercial Pick/MultiValue product set. This new release incorporates significant feedback from many successful MultiValue migrations from other MV platforms over the last few years, as well as introducing over 50 new features and enhancements since V 14.0.

The key significant features cover:

- 64bit delivery across all platforms – providing improved platform compatibility, performance and larger address space.
- Outgoing Web Services – you can now expose Reality applications outbound as well as inbound.
- 64bit SQL/ODBC drivers - Windows and UNIX platform drivers are provided, as well as examples of open Driver Managers for all platforms. Now providing greater levels of conformance for SQL, including whole row data set access rather than individual rows.
- Significantly Enhanced Database I/O performance on

MS Windows platforms.
- Easy migration from other MultiValue Platforms, including definable MV Environments and viable use of Case Insensitivity.

Plus, over 50 feature enhancements released since V 14.0, covering DataBasic runtime and debugger, TCL Macros and command logging, Global Logon & MD, Indexing and general features. ... providing a future proofed MultiValue platform from 1 to many 1,000's of users on standalone to fully resilient and disaster recovery based systems. From start-ups to global enterprises, Reality can be deployed by end customers or using the latest in remote managed services, either housed centrally or within the cloud.

Other Key Features of Reality:

While maintaining the original virtual machine concept, Reality has undergone 40 years of development based on the future needs of both our existing and growing user base. Interoperability using integrated ODBC/JDBC SQL (including accessing external SQL sources and exposing Reality as a SQL database), Web Services, XML Parsing, HTML Interfaces via DataBasic & English, TCP/IP Sockets. Remote calls can also be made in order to provide integration with open Java applications, Microsoft Visual Basic, C/C++/C# and .net.

Advanced User Documentation is included in Reality V 15.0, authored using the leading industry standard MadCap Flare suite that provides for ease of use and is compatible with most browsers. The exampled content is viewable from the product delivery

image, with latest updates always on the Website. One-click feedback can be sent via email in order to help maintain the highest quality content.

Reality Website: www. northgate-is.com/ reality

A new website compliments V 15 providing the latest product updates and news from within Northgate IS, our partners and end users. With the many growing number of business, general IT and MultiValue newsgroups and social media feeds, the Website aims to pull together all key items in order to enable us to remain current with what matters to our resellers and customers.

About Northgate & Reality

Reality is available from Northgate Information Solutions, a privately owned Kohlberg Kravis Roberts global company, and its selected resellers. For further product and consultancy information, including migrating to Reality and contact details refer to the Website at www.northgate-is.com/reality ∎

New Mobile E-Commerce Solution Supports All Mobile Devices

Kore Technologies (www.kore-tech.com), a leading provider of enterprise integration and e-commerce Web solutions for

MultiValue applications and Microsoft SQL Server databases, today announced the general availability of KommerceServer Mobile Edition for its KommerceServer eCommerce Suite.

The KommerceServer eCommerce Suite is Kore's integrated e-commerce web storefront and customer information portal solution designed for manufacturers, distributors, and retailers with support for business-to-business (B2B) and business-to-consumer (B2C) environments.

KommerceServer Mobile Edition includes the advanced features of the KommerceServer webStoreFront and webPortal solutions in the KommerceServer eCommerce Suite, and now provides this same functionality in a compact, easy-to-use mobile version that has been designed and optimized for the requirements and workflow of the mobile user.

"We recognize that many of our customers are distributors serving contractors, designers, and technicians that are often times on a job site without access to a PC browser. KommerceServer Mobile Edition gives our customers that competitive edge in their industry. Kore is committed to leading its own competition by providing the very best mobile features and workflow for those users on the road" said Ken Dickinson, managing partner and co-founder at Kore Technologies

Highlights of the new product include:

- One mobile website supports all mobile devices: iOS, Android, Windows Phone

- Mobile website designed and optimized for the smaller mobile screens
- Alternate product images sized for mobile website speeds up response times
- Automatic mobile device detection redirects to mobile website when browsing
- Search Engine Optimization configuration and parameters extended to mobile website
- Sell all of your products or just a subset on the mobile website
- Enables users to access their account and order history via the customer portal
- Seamless near-real time integration to supported back office systems
- Centralized managment of all products and promotions
- Easy to customize, configure and style

For additional information regarding the KommerceServer eCommerce suite or KommerceServer Mobile Edition please contact Kore Technologies at 866-763-5673 or send an email to sales@koretech.com.

About Kore Technologies

Kore Technologies is a leading provider of enterprise integration, business intelligence and eCommerce Web solutions for MultiValue and Microsoft SQL Server databases. Founded in 1999 as a professional services and software development company, Kore specializes in providing business-to-business connectivity and best-in-class product integration with its flagship Kourier Integrator. Leveraging this technology, and Kores extensive experience with enterprise applications and database systems, Kore delivers solutions for integrating disparate applications, building data warehouses, and developing business intelligence solutions, and creating integrated eCommerce applications. ■

F.W. Davison & Company Embeds Informer's Agile Business Intelligence Capabilities into its Software Platform

A recent benchmark report entitled "Agile Business Intelligence" produced by market research firm Gleanster evaluated 32 business intelligence vendors (BI) in four categories: Ease of Deployment, Ease of Use, Features and Functionality, and Overall Value.

Drawing from the experiences of 367 companies, this 38-page report offers a comprehensive look at how Top Performers are using agile BI to enable continuous, on-demand reporting, empower non-IT professionals with data access and drive faster, smarter business decisions.

According to Gleanster, Informer Partner F.W. Davison is a great example of an organization reaping the benefits of agile business intelligence.

F.W. Davison & Company is a leading provider of high-performance software solutions for professional employer organizations (PEOs). The company sells its software to human resources (HR) outsourcing vendors, some of which have hundreds of thousands of small business clients.

When it comes to employee payroll and benefits, there is no shortage of data. For decision makers, it seems they can never generate enough reports from the data. To help satisfy their need for information and insights, F.W. Davison has not only embedded agile BI capabilities into its platform but has also developed over 1,200 core report templates.

According to Craig Babigian, who serves as executive vice president, that number still falls short of the ideal. "If we had 12,000 templates," he says, "it still wouldn't be nearly enough."

That's because virtually every HR department wants reports in different formats and in different combinations, whether the focus is payroll analysis, employee benefit analysis, or any other type of analysis. That is where the flexibility of agile BI comes into play. "The possibilities are endless in terms of how data can be combined and sorted," says Babigian. "There's not much you can do with a printed report. Now users can run analytics and create their own charts based on their specific needs."

The real beauty lies in the fact that users can combine data from multiple databases into a single report. For example, they can generate a report on payroll information and benefit coverage in one fell swoop, even through the requisite data may come from two separate databases.

Business users are empowered to engage with that report on the screen in multiple ways. They can run analytics that sort, aggregate and summarize certain columns or segments of information. They can schedule tasks to automatically create and email the format. "It's all the same data," says Babigian, "but everyone wants to see it differently."

"Agility is a great word to use in the context of BI when you're trying to create an alternative to the traditional, full-scale approach to data analysis," says Jeff Zabin, Gleanster's research director and co-author of the report. "It speaks to the acceleration of decision-making cycles and the need to put the right data and tools in the hands of business users."

"Informer is a great example of a solution that is bringing the promise of Agile BI to life," says Zabin.

For a free copy of the report, visit http://www.entrinsik.com/informer-gleanster-agilebi.

Want a demonstration of Informer or a free trial? Call us today at 888-703-0016 or email sales@entrinsik.com. ■

LOCKING – PART 3: MODERN LOCKING

Recognizing that an intervening update has taken place is, of course, only the first step. Look what happens when you do some conflicting changes using the Data-Set above (fig. 4).

Not very user friendly, but an improvement on previous releases that just threw an error to the user that their updates were not permitted. This is optimistic locking in the raw: seeing that the underlying data has changed and throwing out the update.

Conflict Resolution

Simply abandoning an update because a conflict has taken place is lazy, unacceptable, and likely to annoy your users. So is reloading the current version of the record and asking them to reapply any changes (assuming they remember them all) which is the other lazy form of optimistic programming.

Good optimistic locking, whether against SQL or MultiValue, is all about merging changes. This means re-applying all those changes made by the user (or process) to the current copy of the record that they made to the original copy, and ensuring that the record is pessimistically locked whilst those updates are applied (or you may just end up in the same loop again).

Back in the first article, we showed what happens when locking fails and two users (Clare and Gary) tried to update the same stock quantity. Because both had open copies of the same record, one of the updates was overwritten.

In a merge locking model, the update to the record would be noted as an action — in this case, adding or removing units from a stock total. The actions are then applied at the point when the records are saved to leave the record in a consistent state. Here's the original timeline rewritten to use merge processing: instead of Gary writing back the stock record only his change in volume is applied over the top of Clare's changes to leave the correct value (fig. 5).

It is important to remember that this does not mean simply applying the same data

updates, but the same processing that led to those updates. Any derived values will need to be derived again using the current rules and data to ensure consistency.

What this really means is a separation of duties. Where traditional MultiValue programming may scatter updates through an entry screen or in flight during user interactions, with merge locking it is essential that the new data is serialized to be applied at the point of committing.

That, unfortunately, usually means code — and possibly lots of it, which is something to consider when budgeting for developing a web based application. Code spent in conflict resolution or segregated into commit processing can be verbose and costly, and is a major factor in the delivery of disconnected applications. **IS**

BRIAN LEACH is an independent Multi-Value and .NET consultant working in the UK. He is past president of the International U2 User Group. Find out more at www.brianleach.co.uk.

```
this.tableAdapterManager.UpdateAll(this.datetestDataSet1);
```

⚠ **DBConcurrencyException was unhandled**   ✕

Concurrency violation: the UpdateCommand affected 0 of the expected 1 records.
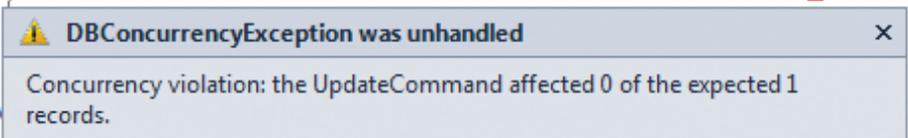
Fig. 4 Concurrency Exception
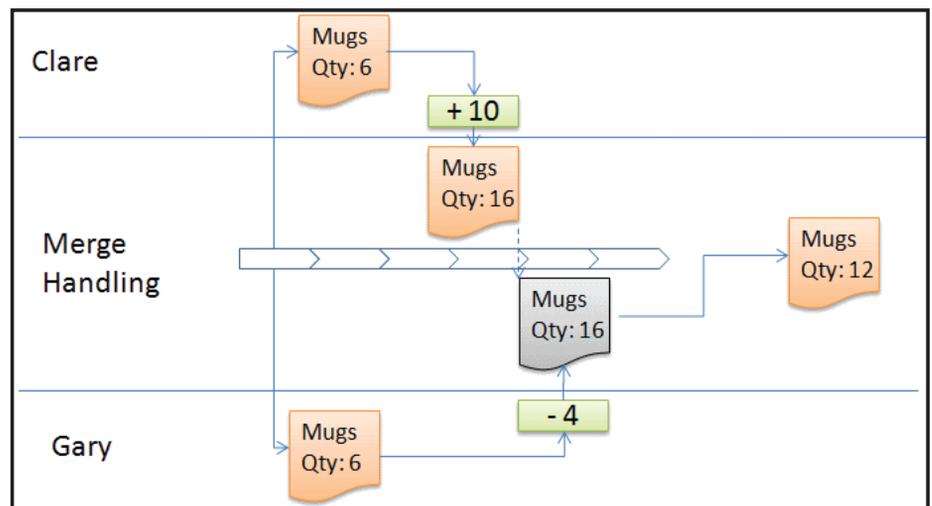
Fig. 5: Merge Processing

The (MultiValue)
Dating Game

BY KEVIN KING

No matter the application, date values typically play a crucial role. When was the order created? When did the packing slip get printed? When will the payment be due?

Fortunately, calculating dates in MultiValue Basic is really pretty easy. Start with an internal date and add or subtract a certain number of days, and output convert the result and we're there, right? While this is a great start, we can do much more with just a little more code.

Let's take a peek at some common date calculations and how we might implement them as MultiValue Basic subroutines.

Calculating the First Day of Next Month

Let's start with something simple. Knowing the current month and year, getting the start of the next month is as simple as adding 1 to the current month and getting the corresponding date. About the only complexity here is watching out for end of year; if the month goes larger than 12, we need roll the month over to 1 and increment the year. See figure 1.

In all of our examples, the TODAY variable represents a day in a month — any day in that month — and the NEXT. DATE variable is returned with the result. The return value is listed as the first formal argument so that these subroutines can be

called both from other Basic routines as well as SUBR(..) type dictionaries for those platforms that support this feature.

Note the NEXT.MONTH -= 12 when the month is greater than 12. In this example this is a little bit of overkill. But stay tuned, this will become an important detail shortly.

Calculating the Last Day of the Current Month

The last day of a month is typically considered one of the more vexing calculations, as you have to factor in the number of days in each month, leap years, and all that, right? Not at all! Unless the Mayans were right, the last day of a month is always followed by the first day of the next month, so all we have to do is calculate the first day of the next month (fig. 1) and subtract a day. See figure 2.

Calculating the First Day of Any Future or Past Month

The calculation of any first day of a month — future or past — requires only minor changes to our first example. Instead of adding 1 to the month, we simply add a number of months future or past. To handle both future and past in the same routine we need to watch for the month being less than 1 or greater than 12 and adjust it accordingly, considering that we may have to count back or forward multiple years. For future years, every time we subtract 12 from the month, we add 1 to the year. For past year, every time we add 12 to the month, we subtract 1 from the year. See figure 3.

Calculating the Last Day of Any Future or Past Month

With what we've created so far, calculating the last day of any future or past month is easy. We simply calculate the first day of the month following the month requested and subtract one day. Also note that by adding 1 to the MONTHS variable in the subroutine argument, the MONTHS variable will remain unchanged when this routine returns to its caller. See figure 4.

Calculating the First Tuesday in a Month

Now that we know how to calculate the first day in a month, it might be useful to be able to calculate the first Tuesday (or any day of week for that matter). Once we know the first Tuesday, we can easily calculate the second and subsequent by adding in 7 (days) for each week.

At first blush this seems like this could get quite complicated. Au contraire! We simply need to figure out the first day of the month (figure 3), back up to the last Sunday, and then add in the value for the day of week we want. Finally, if the final day is before the beginning of the month, we simply add 7 days to push our result into the current month. See figure 5.

There is one UniData quirk that may not exist in other platforms: Sunday could be represented as 0 or 7, depending on how an option is set. Therefore, we have a couple lines in here to change the day of week to 7 if the "DW" (day of week) conversion returns 0.
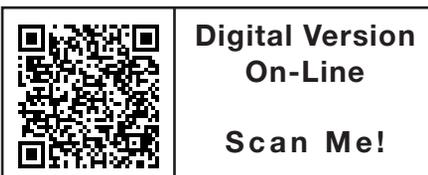
The last day of a month is typically considered one of the more vexing calculations, as you have to factor in the number of days in each month, leap years, and all that, right? Not at all!

Calculating the Last Tuesday in a Month

Calculating the last Tuesday in the month is effectively identical

```
SUBROUTINE FIRST.DAY.NEXT.MONTH(NEXT.DATE,TODAY)
*
TODAY.MONTH = OCONV(TODAY,'DM')
TODAY.YEAR  = OCONV(TODAY,'D4Y')
*
NEXT.MONTH  = TODAY.MONTH + 1
NEXT.YEAR   = TODAY.YEAR
*
IF (NEXT.MONTH GT 12) THEN
   NEXT.MONTH -= 12
   NEXT.YEAR  += 1
END
*
NEXT.DATE=ICONV(NEXT.MONTH:'/01/':NEXT.YEAR,'D')
*
RETURN
```
Fig. 1

```
SUBROUTINE LAST.DAY.THIS.MONTH(NEXT.DATE,TODAY)
*
CALL FIRST.DAY.NEXT.MONTH(NEXT.DATE,TODAY)
NEXT.DATE -= 1
*
RETURN
```
Fig. 2

```
SUBROUTINE FIRST.DAY.ADJ.MONTH(NEXT.DATE,TODAY,MONTHS)
*
TODAY.MONTH = OCONV(TODAY,'DM')
TODAY.YEAR  = OCONV(TODAY,'D4Y')
*
NEXT.MONTH  = TODAY.MONTH + MONTHS
NEXT.YEAR   = TODAY.YEAR
*
BEGIN CASE
  CASE (NEXT.MONTH GT 12)
    LOOP
    WHILE (NEXT.MONTH GT 12) DO
      NEXT.MONTH -= 12
      NEXT.YEAR  += 1
    REPEAT
  CASE (NEXT.MONTH LT 1)
    LOOP
    WHILE (NEXT.MONTH LT 1) DO
      NEXT.MONTH += 12
      NEXT.YEAR  -= 1
    REPEAT
END CASE
*
NEXT.DATE=ICONV(NEXT.MONTH :'/01/' :NEXT.YEAR,'D')
*
RETURN
```
Fig. 3

```
SUBROUTINE LAST.DAY.ADJ.MONTH(NEXT.DATE,TODAY,MONTHS)
*
CALL FIRST.DAY.ADJ.MONTH(NEXT.DATE,TODAY,MONTHS + 1)
*
NEXT.DATE -= 1
*
RETURN
```
Fig. 4

to the earlier example, except that we start from the last day of the month, back up to Sunday, add in the day of week we want, and if it's beyond the end of the month we subtract 7 to pull the result into the correct month. See figure 6.

Calculating a Week Number

Being able to calculate a week number can be a very useful thing, especially for financial and metric reporting. However, this calculation is often victim of interpretation. For example, what exactly constitutes a "week"? Is it Sunday through Saturday? Wednesday through Tuesday? Or does the first day of the week depend on the day of week of the first day of the year?

When the first day of the week is based on the first day of the year, calculating a week number is simple; subtract the current date from the day before the first day of the year, divide by 7, take the integer result, and add 1 if there's any remainder from the division. Though the calculation is simple, it suffers under the problem that in a Monday through Friday work week some days could be assigned one week number, whereas other days in that same work week might be assigned a different number.

To try to keep Monday through Friday together, let's calculate the week number based on Sunday to Saturday. We start by calculat-

```
SUBROUTINE FIRST.MONTH.DOW(NEXT.DATE,TODAY,DOW)
*
FIRST = ''
CALL FIRST.DAY.ADJ.MONTH(FIRST,TODAY,0)
FIRST.DOW  = OCONV(FIRST,'DW')
IF (FIRST.DOW EQ 0) THEN
   FIRST.DOW = 7
END
*
NEXT.DATE = FIRST - FIRST.DOW + DOW
IF (FIRST.DOW GT DOW) THEN
   NEXT.DATE += 7
END
*
RETURN
```
Fig 5

```
SUBROUTINE LAST.MONTH.DOW(NEXT.DATE,TODAY,DOW)
*
LAST = ''
CALL LAST.DAY.ADJ.MONTH(LAST,TODAY,0)
LAST.DOW   = OCONV(LAST,'DW')
IF (LAST.DOW EQ 0) THEN
   LAST.DOW = 7
END
*
NEXT.DATE = LAST - LAST.DOW + DOW
IF (LAST.DOW LT DOW) THEN
   NEXT.DATE -= 7
END
*
RETURN
```
Fig. 6

THE (MULTIVALUE) DATING GAME

ing January 1 of the current year, and then back it up to Sunday, understanding that the Sunday may actually be in the prior year. Subtracting today from that Sunday tells us how many days have elapsed, and dividing this by 7 and taking the integer result tells us the number of weeks. Well, almost. If there is any remainder left over from the division, or if the week number is 0 (as it would be when January 1st falls on a Sunday), we add 1 to the result. See figure 7.

Calculating This Day in a Future or Past Month

Interestingly, this is one of the more complex date calculations. In short, while the 15th is valid in every month, the 31st is not. In this latter case we need to roll the 31st back to the 30th, or 28th, or even the 29th every leap year. Fortunately, we don't have to worry too much about all those details. As this only happens at the end of a month, we can attempt to calculate a date for a given month, day, and year, and if it's not valid, we can decrement the day number and try again until we get a valid date. See figure 8.

Determining U.S. vs. International Date

In all of the examples so far I've assumed that the dates will be formatted as month/day/year, as they are here in the U.S. Certainly it would be no big deal for our non-American friends to translate these routines into the day/month/year format used by much of the rest of the world. But what if you're writing an application that needs to support **both** U.S. and International date formats?

Most MultiValue systems have some system variable that will tell whether the system is configured for U.S. or International dates. While I appreciate that such a feature is available, I've never quite grown to trust it, especially not for application portability. Besides, making this determination in code is really quite simple.

If we try to calculate a known date in U.S. format and the ICONV(..) returns nothing, we can have some confidence that the international date format is enabled. Nothing more complicated is needed. See Figure 9.

In Summary

Calculating dates is an important feature for just about every application. And compared to working in other languages, MultiValue databases make date calculations easy. Certainly there are other more involved calculations but hopefully this has shown that the whole process doesn't have to be all too complicated. **IS**

KEVIN KING is the President and Chief Technologist with Precision Solutions, Inc., a leader in technology solutions, support, and training. He is also the author of SB+ Solutions, an enthusiastic private pilot, and Christian guitarist and producer... as time allows.

```
SUBROUTINE WEEK.NUMBER(WEEK,TODAY)
*
TODAY.YEAR = OCONV(TODAY,'D4Y')
JAN.1      = ICONV('1/1/' : TODAY.YEAR,'D')
JAN.1.DOW  = OCONV(JAN.1,'DW')
IF (JAN.1.DOW EQ 0) THEN
   JAN.1.DOW = 7
END
*
IF (JAN.1.DOW EQ 7) THEN
   SUNDAY = JAN.1
END ELSE
   SUNDAY = JAN.1 - JAN.1.DOW
END
*
WEEK = INT((TODAY - SUNDAY) / 7)
IF (MOD(TODAY,SUNDAY) GT 0) OR (WEEK EQ 0) THEN
   WEEK += 1
END
*
RETURN
```
Fig. 7

```
SUBROUTINE THIS.DAY.NEXT(NEXT.DATE,TODAY,MONTHS)
*
TODAY.MONTH = OCONV(TODAY,'DM')
TODAY.DAY   = OCONV(TODAY,'DD')
TODAY.YEAR  = OCONV(TODAY,'D4Y')
*
NEXT.MONTH  = TODAY.MONTH + MONTHS
NEXT.YEAR   = TODAY.YEAR
*
BEGIN CASE
   CASE (NEXT.MONTH GT 12)
      LOOP
      WHILE (NEXT.MONTH GT 12) DO
         NEXT.MONTH -= 12
         NEXT.YEAR  += 1
      REPEAT
   CASE (NEXT.MONTH LT 1)
      LOOP
      WHILE (NEXT.MONTH LT 1) DO
         NEXT.MONTH += 12
         NEXT.YEAR  -= 1
      REPEAT
END CASE
*
LOOP
NEXT.DATE =ICONV(NEXT.MONTH:'/':TODAY.DAY:'/':
      → NEXT.YEAR,'D')
WHILE (NEXT.DATE EQ '') DO
   TODAY.DAY -= 1
REPEAT
*
RETURN
```
Fig. 8

```
SUBROUTINE DATE.FORMAT(ANS)
*
IF (ICONV('02/28','D') EQ '') THEN
   ANS = 'I'
END ELSE
   ANS = 'A'
END
*
RETURN
```
Fig. 9

CLIF NOTES: IT'S GOING TO GET BUSY

application system development, there are many fine solutions, from people who understand us and our data model, to explore and use to increase our productivity in getting the job done. (Take a look at the back issues of the magazine for both advertisements and announcements in our Press Room section.)

This is by no means an exhaustive list of the things a modern MultiValue developer has available to them. These were just a few of the ones that particularly interested me as I was musing on how our MultiValue industry keeps changing. I'm sure that there are some technologies not mentioned that have caused your nose to start twitching in anticipation. I'd like to hear your thoughts. Send your comments to editor@intl-spectrum.com.

As for me, I'm tired just thinking about how many things there are to pursue in 2013. I think I'll go take a nap. **IS**

Feedback

What came first, the letters or the letters-to-the-editor department?

International Spectrum Magazine has a Feedback Department, sometimes known as Letters to the Editor.

We want to hear your comments, your reactions, your agreement or disagreement with what you see. Also, do not hesitate to let us know about things happening in the MultiValue Community we may not have heard about yet.

Please send your comments by e-mail to:
editor@intl-spectrum.com

MARKETPLACE

HIGH IMPACT 4-COLOR DISPLAY AD

Data Systems, Inc.
Sophisticated order processing and fulfillment systems

Data Systems, Inc.

303.333.3333   info@example.com   www.example.com

3 -3//8"   1"
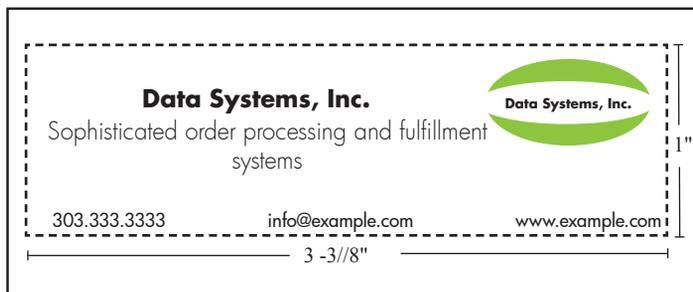
Cost Per Year

1" $1,811.00

2" $3,623.00

BASIC LISTING: $690.00

DataSystems, Inc. http://www.example.com, ... 333.333.3333, ext. 1

- Includes company name, website, and phone number
- Approximately 60 characters per line
- Place under category of your choosing

ADDITIONAL LINES: $280.00 per line

DataSystems, Inc. http://www.example.com, ... 333.333.3333, ext. 1
Sophisticated order processing and fulfilment systems

- Approximately 60 characters per line

ONLINE BANNER ADS:

VERTICAL BANNER (120X240)

$2,880.00 per year or $300.00 per month

RECTANGLE (180X150)

$2,592.00 per year or $270.00 per month

SQUARE BUTTON (125X125)

$1,680.00 per year or $175.00 per month

E-MAIL NEWSLETTER BANNER:

HALF BANNER (300X90)

$4,200.00 per year or $175.00 per issue

ONLINE MARKETPLACE LIST:

BASIC LISTING – FREE

- Company Page
- Product Page – Limit 1
- ADs will be displayed on your page

PROFESSIONAL LISTING

$350.00 per year/ $35.00 per month

- Company Page
- Product Pages - Unlimited
- No ADs displayed on your page
- Download Links

For more information contact Nathan at: nathan@intl-spectrum.com

ACCOUNTING

Natec Systems
www.natecsystems.com | nrector@natecsystems.com

COMPLIANCE

SJ+ Systems Associates
www.sjplus.com | sjoslyn@sjplus.com

CONSULTING

Clifton Oliver & Associates
www.oliver.com | wco@oliver.com

Drexel Management Service
www.drexelmgt.com | dconboy@drexelmgt.com

Execu-Sys, LTD
www.eslny.com | mh@eslny.com

HDWP
www.HDWP.com | results@HDWP.com

PICK Programmers Shop
www.pickprogram.com | brian@pickprogram.com

Precision Solutions
www.precisonline.com | Kevin@PrecisOnline.com

DATABASE

Ladybridge Systems Ltd
www.ladybridge.com | sales@Ladybridge.com

Revelation Software
www.revelation.com | info@revelation.com

REPORTING

Brian Leach Consulting, LTD
www.brianleach.co.uk | brian@brianleach.co.uk

Entrinsik
www.entrinsik.com | sales@entrinsik.com

TERMINAL EMULATOR

AccuSoft Enterprises
www.asent.com | pjs@asent.com

WANT TO SEE A SPECIFIC TOPIC?

International Spectrum is looking for writers, feedback, and topic ideas. We all have specific topics and issues that we need answers to find solutions for. Send us an E-mail with topics you would like to have covered in the magazine or on the website.

E-mail: nathan@intl-spectrum.com

WANT TO WRITE?

Expand your professional credentials, and provide us with an article.

Give us a rough and ugly outline, and we will help you refine it, proof it, and make it press ready. Or you can give us something polished, proofed, and press ready to publish.

Share your thoughts and expertise with over 10,000 fellow MultiValue developers and users.

E-mail: editor@intl-spectrum.com

LETTERS TO THE EDITOR

Have an opinion on an article: Agree, disagree, or enhancement to an article from a previous issue? International Spectrum and our authors are interested in hearing from you!

E-mail: editor@intl-spectrum.com

NEED A MENTOR?

Mentors give developers the ability to ask industry experts for direction, code examples, and/or just ask them to see if something makes sense. Sometimes, all you need is a resource or example to start or complete a project.

Check with us to see who is available for mentoring, and how you can take advantage of it to save your business or company money.

E-mail: nathan@intl-spectrum.com

WANT TO BE A MENTOR?

We have many retired or semi-retired professionals out there that would love to share their knowledge of MultiValue development. If you are one of them, please contact us to see what mentoring is all about.

E-mail: nathan@intl-spectrum.com

Our Apologies...

LadyBridge Systems Ltd end of the year recap responses were missed in the article originally printed in the Nov/Dec 2012 issue. Here are their reponses:

What new features or services did Ladybridge Systems Ltd introduce in 2012?

We released Unicode character set support in April, making QM more attractive to developers targeting markets in which the 8-bit character set is not sufficient. In accordance with our forward compatibility policy, existing programs continue to run unchanged and without recompilation, even when accessing files that contain data from the extended character set supported by this development. Extended character set support within the QM environment is coupled with 8-bit encoding options such as UTF-8 on external interfaces to terminal devices, directory files, etc.

Flexibility is provided by user editable character maps that determine character type attributes, upper/lower case pairing, sort order, transliteration and other features. We worked with our business partners and beta test sites to ensure that we had character maps for Dutch, English, French, German, Italian, Polish and Spanish at first release and we are always keep to receive maps for other languages. Because the character map is selectable separately for each QM session, different users of the same report might see the data sorted differently for their local language conventions.

What was the greatest or most exciting thing that happened to your company or product in 2012?

Despite the global recession, QM continues to grow and has attracted serious interest from developers of several large applications looking to lower costs and expand markets. We have forged several new business partner relationships during the year, notably with Revelation and Ashwood Computer. The QM connector for Revelations OpenInsight and O4W was launched in March and combines the power of Revelation's tools with the low cost and reliability of the QM multivalue database.

The mvTools package from Ashwood Computer was ported to QM in September and provides a GUI application development environment that is closely linked with the AccuTerm terminal emulator. **IS**

CLIF NOTES: IT'S GOING TO GET BUSY

little device that they carry with them everywhere (even into the restroom, for heaven sakes!) And the requests of the mobile market exploded. Since smartphones come with web browsers built-in, websites are now expected to be "reactive," meaning that they automatically detect what kind of devices "are" using what kind of browser to view them and provide a customer experience that is tailored to that platform. Fortunately, designing reactive websites is becoming a little bit easier through the use of "platform-aware" content management systems such as Joomla!, Drupal, and WordPress, to name just three of the more popular frameworks. (As a side, or snide, note, I was always amused that the Apple website was not iPhone aware and was virtually unreadable on the device that took the mobile market by storm at the time.)

But websites alone are not enough. Sometimes the desired user experience requires a bit more than web technologies can provide, or there is a need to capture, process, and store data when no web connection is available. For that, we have Apps.

Ooh, look. We have yet another box of goodies to wallow in. There are frameworks available to help ease the workload of producing mobile apps, such as Adobe Cordova, formerly known as PhoneGap. Or, you may decide that you need to get down into the guts, and do good old fashion programming. Typically that requires learning Objective-C if you're going to develop for the iPhone SDK or Java if you want to develop for Android. But even that is changing. For example, using a product like MonoTouch for "iOS" and Mono for Android you can write cross-platform applications for these mobile devices using C# on .NET. I imagine there are others I haven't run across yet or are lurking just offstage ready to be introduced.

Let's not forget our own of beloved MultiValue Tools Vendors. From attractive reporting to advanced Business Intelligence to system connectivity to full scale

It's Going to Get Busy

BY CLIFTON OLIVER

The year 2013 is barely underway, and I already feel like I'm falling behind. What a great time to be a MultiValue developer. There are just so many new ~~toys to play~~ with technologies to explore that I hardly know where to start. Unless a person is a MultiValue couch potato, clinging to their outdated skill set and hoping to just slide by without learning anything new until they can retire, how could anybody be bored? With all of the major MultiValue platforms now having good connectivity to the latest modern technologies, there are numerous ~~adventures~~ opportunities to move the user experience off of the database and let MultiValue do what MultiValue does best — model data, store and retrieve it efficiently, and provide the application developer with a cost-effective database that is flexible, easy to work with, and doesn't insist on getting in the way of what you're trying to accomplish. Here are a few examples of what I'm talking about.

First up is the topic of GUIs. I know very few MultiValue developers who have not stopped attempting to argue that green screens are preferable to GUIs. The users, and just as important to us developers, the people who write the checks to buy stuff from us, have spoken. And with a loud voice and a heavy wallet have declared that Green Screen is Dead. Of course, with the thousands of green screen programs still in production, nobody with any experience would consider suggesting rewriting all of them in a massive "do or die" replacement project. But in most cases, any MultiValue developer who refuses to consider doing all new development (and possibly an as needed replacement on a program-by-program basis) in some form of modern GUI is just begging for a job transition to being a greeter at Walmart.

So how to put GUIs on our applications; what to use? There are number of ways of doing it, of course. Look at just a few of your options. You can go with some form of GUI-based application development tool provided by your MultiValue provider. Or you could get down and dirty with something like Java and Swing. Or you could use the NetBeans IDE and the Swing GUI Builder. Not fond of Java? Well there's always the old standby, Visual Basic. In its latest incarnation, Visual Basic.NET is a pretty easy language for MultiValue Basic programmers to transition to. And the.NET framework does most of the heavy lifting for you. If you feel too pure to sully yourself with Visual Basic, there's always C #.NET. And now with an open source package called Mono, the .NET framework is available on Linux, Mac, and Unix.

There are number of techniques and products for application development with rich GUI interfaces that make use of intermediate web servers and web technologies.

And then came the demand to make data available to the users via the web (both Internet and intranet). My, oh my. What a wonderful box of gizmos we have to rummage through here. Of course, there are the new versions of HTML, XHTML, and other variants. If you took a look at HTML a few years ago in its early forms and turned your nose up at it thinking it was little more than a glorified RUNOFF, you better look at it today. There is CSS, Cascading Style Sheets, which is intended to allow you to separate the design and appearance of a website from the actual content. And as you would suspect, any good idea like CSS will eventually start popping up in other uses than what it was originally developed for, such as documentation preparation and formatting from MultiMarkdown to La-TeX or PDF.

Web programming, anyone? There are languages such as JavaScript, PHP, Python, Ruby on Rails, and many others (including a bunch I'm not even aware of, I am sure). And to aid in simplifying the coding of the customer experience, you have such tools as JQuery and JQuery mobile. (You may want to check the archive of back issues of International Spectrum magazine. Kevin King has written several articles about some of these web technologies and how to tie them in with MultiValue.)

Then came the smart phones, and the user demand for access to data by that