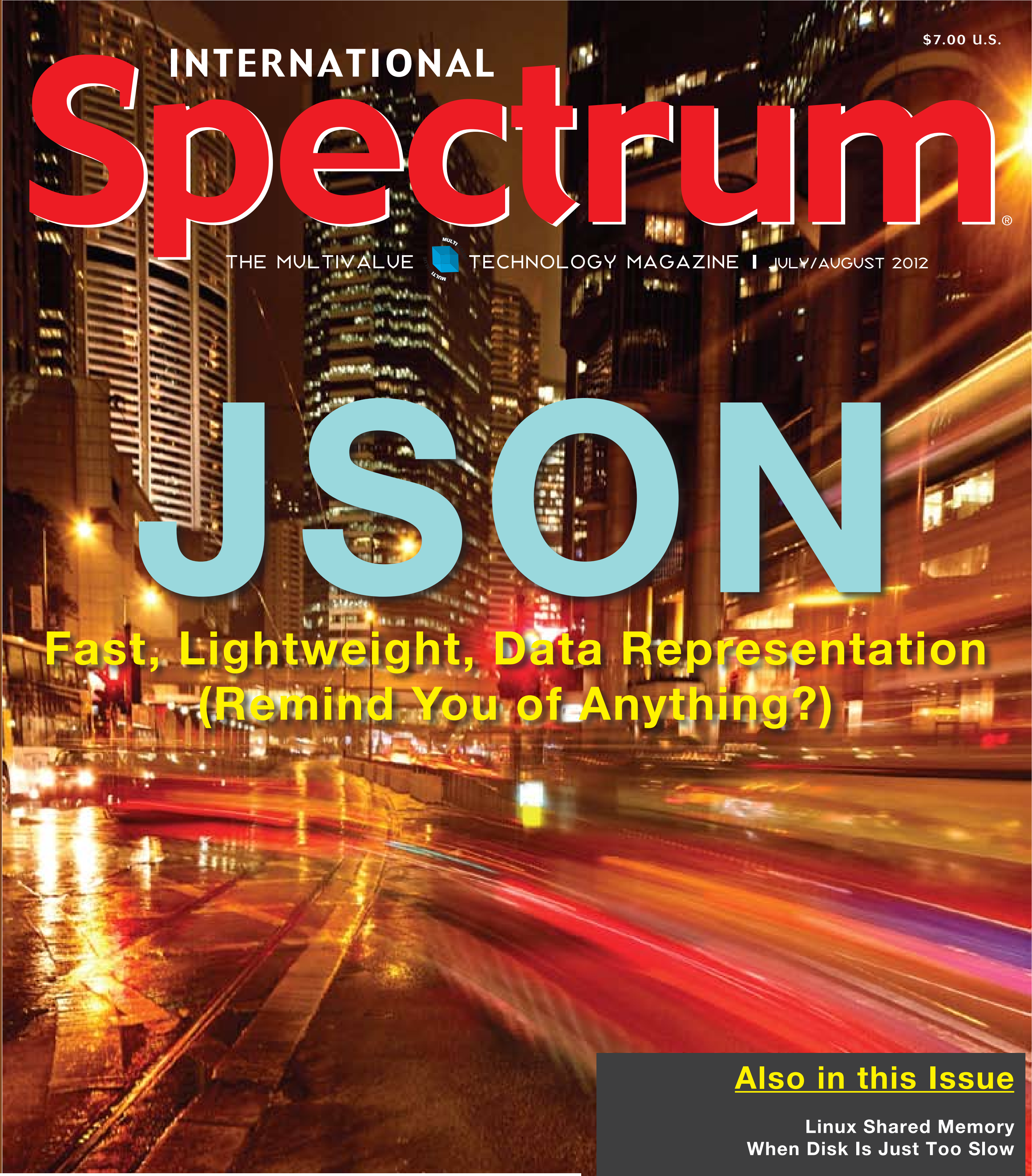
$7.00 U.S.

INTERNATIONAL Spectrum®

THE MULTIVALUE TECHNOLOGY MAGAZINE | JULY/AUGUST 2012



# JSON

## Fast, Lightweight, Data Representation (Remind You of Anything?)

## Also in this Issue

intl-spectrum.com

The ideal database system
would provide
object and relational
views of data.

InterSystems
CACHÉ®

Download or request a free, fully functional, non-expiring trial version at **InterSystems.com/Ideal1WW**

INTERNATIONAL Spectrum®
THE MULTIVALUE TECHNOLOGY MAGAZINE

## FEATURES | JULY/AUGUST 2012

## DEPARTMENTS

# From the Inside

Technology is making the office obsolete, but are IT and the enterprise keeping up?

Offices are almost not needed in today's working environment as the Internet, Virtual Desktops, and Cloud applications can been found and accessed from anywhere. Skype is now part of the office space, virtually guaranteeing every employee can attend a meeting. Productivity, costs-analysis, day-to-day monitoring, and budget building can all be performed on a computer anywhere so long as it can link to a collective server.

Not to mention the fact that just about every working person has an iPhone, Blackberry, Samsung, or HTC Android phone — or some other model of smartphone. What once use to be laptops, cell phones, and daily trips to the office, is now becoming smartphones, tablets, video conferencing, and e-mail, then spending the day on site or doing sales calls.

Even in warehouse production jobs, more and more of the manager's time is spent on the floor than in the office doing paperwork. Automation and the paperless office has always been the goal of business, and new technology has been pushing that fast and hard, not to mention being more successful at it.

While many professionals, mainly IT, question the reliability of cloud computing and the consequences of relying too much on it, the current IT spending is moving further and further from the traditional office. But the enterprise software development, security, and infrastructure is not.

While existing management styles focus on the office as a place to control productivity and need the IT infrastructure to support it, the day-to-day work is progressing outside those controls. That is upsetting management and IT.

Imposing the controls that used to work in a normal office structure to a mobile or virtual environment does not work, as many businesses are finding out. While some businesses try to control the mobile environment by controlling the hardware, more and more IT experts are suggesting using application and software controls, leaving the hardware and UI interface in the employee's hands. (See my last FTI – BYOD)

Now back to my original sentence... are the enterprise and business applications ready for this? No, I don't believe we are. Too many of our systems have been built around the office and the controls and security that the office provides, and less about using the data and software in a virtual and mobile environment.

If we don't go mobile and change our systems to provide controls and security through the application, not the hardware, our applications are going to fail. Everyone else outside the MultiValue space calls this Cloud Computing; I like to call it... business as usual. But many of our applications are built around the desktop or desktop UI.

Those have to change. If they don't change, IT beware!

Workers are expecting you to provide them their data when and where they want it, and now know enough to know that it is possible. iTunes and the Google Play has succeeded in making our job harder by setting user expectations.

Innovate your MultiValue Application or your MultiValue Application will die! If you don't know how, then you need to plan on join us at the 2013 conference. There will be a lot of knowledge delivered about Mobile apps (iPhone, Android, Microsoft Metro, etc.) and how to interface them with the MultiValue application.

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

twitter.com/intlspectrum

intl-spectrum.com/facebook

INTERNATIONAL
Spectrum
JULY/AUGUST 2012

NATHAN RECTOR
*President*

CLIFTON OLIVER
*Editor*

TRACEY RECTOR
*Layout*

*Learn more about the MultiValue Symbol and see what MulitValue Technologies and MultiValue Communities exist to help you support and manage your business and systems. To find out more visit*

**http://www.intl-spectrum.com**

**MISSION STATEMENT** *International Spectrum* magazine's editorial mission is to be the premier independent source of useful information for users, developers, and resellers of MultiValue database management systems, open systems business database solutions, and related hardware, software, and peripherals. Published bimonthly, International Spectrum provides comprehensive coverage of the products, companies, and trends that shape the MultiValue marketplace as well as the computer industry at large — helping its readers get the most out of their business computer systems.

*International Spectrum* is published six (6) times per year at the subscription price of $40.00 U.S. in the U.S.A.; $45.00 U.S. in Canada and Mexico; $50.00 U.S. for other countries. Single copy rates are $7.00 U.S. in the U.S.A. and Canada, and $9.00 U.S. in all other countries. International Spectrum is published by International Specturm, Inc., 8956 Fox Drive #102, Thornton, CO 80260; Tel: 720/259-1356; Fax: 603/250-0664 E-Mail: request@intl-spectrum.com.

PRINTED IN USA

NEWS RELEASES/UNSOLICITED ARTICLES
International Spectrum is eager to print your submissions of up-to- the-minute news and feature stories complementary to the MultiValue marketplace. Black and white or color photographs and diagrams are welcome. Although there is no guarantee a submitted article will be published, every article will be considered. Please send your press releases, articles, and queries to: editor@intl-spectrum.com. International Spectrum retains all reprint rights.

# JSON

## Born for MultiValue

BY KEVIN KING

At a fundamental level, moving information between disparate systems is easy. With a *pathway* (i.e. a network cable or wireless connection) and a *method* (i.e. a TCP/IP network connection or even a drop file interface), the only other thing needed is some form of a *protocol* for structuring the information in transit.

There are a variety of different options for structuring information in transit. On the simple end of things one could create a delimited stream of characters like a MultiValue record with attribute, value, and subvalue delimiters. The advantage to this approach is that it's easy. We, the MultiValue People, know delimited information inside and out. However, this may be a disadvantage for the receiver of the information as they may not share our fondness for delimited fields.

On the more complex side of the equation, we could certainly use something like XML for transmitting structured information. While XML is excellent for representing complex data structures and is well received throughout the wide world of technology, one can't help but feel that XML is the new COBOL. While it's perfectly capable of doing a job, it's verbose to a fault… sort of like the step-uncle who can't stop telling the family of his medical maladies during Thanksgiving dinner.

In the middle of these two options we find something of a sweet spot with Javascript Object Notation, better known as JSON (pronounced JAY-sahn). Using a notation derived from the object syntax of the Javascript programming language, JSON is a lightweight data format that is easy for humans to write and for machines to parse and generate. Like XML, JSON can handle nested data structures with ease and is well received throughout the technology community at large. Yet unlike XML, JSON uses simple characters in representing complex data structures instead of nesting a bunch of start and end tags. This brevity alone allows a JSON message or data structure to be represented with a smaller total character count than XML, thus reducing the bandwidth needed to transmit the message and improving the overall efficiency of the transmission.

To encode something using the JSON format, one must first understand that JSON builds structures with only three types of information:

1. Scalars (single values)
2. Arrays (lists of values)
3. Objects (name: value pairs)

If these first two remind you of an attribute and a MultiValued list, then we're definitely on the same page. However, JSON can represent more than simple attributes and values to allow arrays and objects to include other arrays and objects to an almost unlimited depth.

Let's take a closer look at each of these, shall we?

*JSON is well supported in dozens of contemporary languages, thus opening a door for us to trade our MultiValue information with an ever-widening audience.*

A *scalar* is a simple element, like a number, date, time, or a string. Numeric values do not need to be quoted (using double quotes), but everything else should be quoted, as in:

```
21.25
“06/01/2012”
“15:23:55”
“Hello World!”
```

Multiple scalars can be combined into an *array* by listing multiple items with a comma between each and surrounding the entire list with brackets, like this:

```
[21.25,”06/01/2012”,”15:23:55”,”Hello
     →  World!”]
```

Note how an array may contain disparate types of information, not unlike a MultiValue record. In fact, each item in the array may itself be a scalar, array, or object. By *nesting* arrays and objects inside of each other, structures of any depth or complexity can be represented very easily with JSON. For example, here’s a JSON representation of a simple record with two attributes where the second attribute is a MultiValued list:

```
[“My Company”,[“Bob”,”Sally”,”Waldo”,
 → ”Karen”]]
```

So where’s Waldo? I’m betting you easily found him in the third MultiValue of the second attribute.

JSON may also describe an *object*, where a comma-delimited series of “name”:”value” pairs are wrapped in braces, like this:

```
{“company”:”Precision Solutions, Inc.”,
     →”address”:”630 15th
Ave, #100”}
```

If that all looks a little weird, just think of the names (i.e. “company”, “address”) as dictionary names and the values as the corresponding attribute values. Of course, each value can be a scalar, an array, or an object where the arrays and objects can contain more nested scalars, arrays, and objects.

While the field names do not need to be quoted, it’s usually a good idea to quote them anyway to avoid conflicts for the consumer of the information. For example, certain keywords in Javascript are perfectly suitable field names when quoted (i.e. “new”), but may cause problems for a data consumer if quotes are omitted.

“What about associated multivalues,” you ask? “How might these be represented in JSON?”

To illustrate, let’s say that we have a record with a multivalued product number and a corresponding multivalued field for quantity and another for price, maybe something like this:

```
001 TV]RADIO]LAPTOP
002 1]2]4
003 72999]399]56994
```

In JSON, we have a several options to represent this. First, we could create arrays for each of the multivalued attributes and then combine those into another array, like this:

```
[[“TV”,”RADIO”,”LAPTOP”],[1,2,4],[72999,399,
     → 56994]]
```

While this is a very compact way of representing the information, it may not be the best approach, as it’s more difficult to see associations between fields. Perhaps a better approach might be to create objects for each association and then wrap those objects in an array, like this:

```
[{“product”:”TV”,”quantity”:1,”price”:
     → 72999},{“product”:”RADIO”,”quantity”:
     → 2,”price”:399},{“product”:”LAPTOP”,
     → ”quantity”:4,”price”:56994}]
```

While using objects creates a bigger JSON string, it clarifies that the information is an array (using []) of objects (using {}), where each object has three fields (product, quantity, and

*Continues on page 29*

# Linux Shared Memory and your MultiValue Database

BY JOHN THOMPSON

These days if you are running a MultiValue database, most of your users and executives expect instant access to their data. In the good old days (sometime before I was even doing this technology thing), there was the "Data Processing" department. The Data Processing department took care of the database, often developed new applications, and ran the reports for the users and executives. Today, the users want the ability to run their own reports and "drill down" on the data to be able find what they are looking for easily and quickly. If you find yourself the only MultiValue developer in your company, without a lot of funding, you may be thinking, "Yeah, Yeah, my company will never buy any tool to do that anyway, so what's the point? Plus, I haven't got the time to figure out how to write that kind of a tool."

> *Use shared memory instead of the physical disk to be your transport layer for the data. Yes, this method may not have connection pooling and some other nice features, but because you are using the system RAM, it will still be really fast.*

If you find yourself reading this and remembering the good old days, take heart, because you aren't an old dog. You're just experienced, and you have a lot you can still teach to the new dogs, while learning some new tricks.

Age and history aside though, in the present, let's take a step back for a moment, and consider fundamentally, "What do I need to get my users quicker, instant access to my data?"

At a minimum you need the following:

1. Take input from the user and pass some information to a MultiValue subroutine.
2. Make that MultiValue subroutine give some information back.
3. Display that information in a meaningful way to the user

Yes, the process above is over simplifying the technology you might use underneath to do this, but in reality, this is all you need to talk to the web, the mobile phone, or some other device a user may be using in the year 2012 and beyond.

In its simplest form, the method of passing a MultiValue subroutine some user input and making that subroutine write data out is often done with simple operating system files. If you have ever written a Basic program to export data into an Excel Spreadsheet, you have already done this. If you have been reading Kevin King's series of articles discussing connecting your MultiValue database to the web using the PHP language and the Apache web server, you will notice Kevin is doing much the same thing. With a web server, running an Apache setup alongside your MultiValue database server, you can serve up data on web enabled devices and give users access to it in real time, anywhere they have an Internet connection. Again this is nothing new to some of you.

This business of calling a Basic subroutine with some user input to write data out is very powerful and with enough thought can do some really elegant things besides just sending the user a text file.

However, there is one small downside to doing this on a large system. The naysayers will quickly point the finger and say, “If I have a 1,000 users all calling subroutines on my system and writing data out that is an awful lot of extra disk I/O. Plus, what about connection pooling? I need those responses to be fast and efficient! Why make a session log on and log off every time a request is made? That is really inefficient.”

Well, Mr. Naysayer, if you are still reading, and your MultiValue database server happens to be running any recent flavor of Linux, then I have a proposition for you. The proposition is, use shared memory instead of the physical disk to be your transport layer for the data. Yes, this method may not have connection pooling and some other nice features, but, because you are using the system RAM, it will still be really fast (despite the inefficiencies). So let’s take a look at how to take advantage of Linux shared memory.

By default, included in any Linux operating system with a fairly recent 2.6.x kernel, there is an implementation of shared memory that is very easy to use. The best part is, you don’t have to know any other programming language, other than MultiValue Basic to use it.

Now for the fun part. Get yourself to the Linux shell (preferably with a vt or some supported emulation) and type the following:

```
df -h
```

Look for a line that looks similar to this:

```
tmpfs   16G   0   16G   0%
      → /dev/shm
```

Some of you who know a bit about Linux, are saying, “Wait a minute, you just asked me to display the amount of free disk space on my system?” Yes, that is true, but, what you may be unaware of is that the little line that says tmpfs, is a filesystem mounted in shared memory! This means the mount point /dev/shm is not using the disk!

This is created by default on any Linux install and is mounted every time the machine boots. The maximum space that /dev/shm can grow to is half the system RAM. Even if that space is exceeded, fret not,. It will start using the swap space on the physical disk.

*Continues on page 15*

# Interview with Kelly Bleck of Precision Solutions

We're talking with Kelly Bleck of Precision Solutions. When Kelly first came to our attention, we were intrigued with the idea of someone with no formal training in programming becoming a qualified MultiValue developer. While a lot of of our older contingent started that way in the 1970s, we thought you would enjoy hearing the opinions of someone who accomplished that more recently — in the current millennium, to be precise.

**SPECTRUM:** ***We understand that your background and training is in journalism, not computer programming. Is this correct?***

**KELLY:** Correct — strange I know. I had a little experience with PHP and web design through my journalism college courses, but I had not had any experience with MultiValue or programming per se.

**SPECTRUM:** ***That is quite a bit different than the background of most newcomers to the MultiValue database environment. How did you come to be a MultiValue application developer?***

**KELLY:** I found an employer who was willing to teach. Being that I came from a journalism background, I was essentially a clean slate — no previous coding experience, no "bad habits," no prior biases. So I was very malleable and could be trained to do my job exactly the way it needed to be done.

> *Understanding the client's business model was the hardest to grasp. I wanted to know why people wanted these programs, what the purpose was for them, and how this integration would benefit their business.*

**SPECTRUM:** ***What do you think of the MultiValue database? Did you find it difficult to learn?***

**KELLY:** For me, personally, no. I have always enjoyed problem solving and puzzles, and that is how I have approached learning about MultiValue databases. Everything works together in a very particular way, and new programs can be integrated easily — if you know where and how to integrate them. This discovery and analysis is something I thrive on.

Spectrum: What did you think of the MultiValue data model with everything being variable length and these multivalue and subvalue nested things? Did that seem odd at first?

Kelly: I did think the MultiValue data model was strange at first. But now I think of it as a pretty efficient... storage system, I guess you could say. You're able to have much more information that can relate to a variable number of other pieces of information. It allows for a very wide range of information storage and program availability. Much more fluid and accessible.

**SPECTRUM:** ***And what about the MultiValue Basic programming language? How much trouble did that give you?***

**KELLY:** It was challenging at first, because you are literally learning a new "language." Being a writer I have quickly picked up on the rules — it's like picking up a new writing style. My journalism background has helped immensely with my attention to details, which is necessary when using so many variables in one program. I am a natural editor, so I catch my mistakes (at least I like to think most of the time) and review my programs and applications with a very critical eye. Overall, I've learned it's about repetition; repeating certain phrases, loops, executions that helps me retain the most information.

**SPECTRUM:** ***What would you say was your biggest challenge learning to develop business applications on this platform?***

**KELLY:** Understanding the client's business model was the hardest to grasp. I wanted to know why people wanted these programs, what the purpose was for them, and how this integration would benefit their business. This is a rare thing to gather from your clients — most requirements are bareboned and minimal, leaving a lot to our interpretation. I wanted to be more involved with the clients, which I luckily have found with Precision Solutions.

I guess being an interviewer in my previous life has made me want to be much more involved with those I'm serving.

**SPECTRUM:** ***Do you have any advice for others who would like to learn to develop MultiValue applications?***

**KELLY:** Be open minded. That's the easiest thing to say and the hardest thing to practice. Understand that programming and MultiValue systems are all about rules (at least that's what I've learned), and these rules do not necessarily fit with those you have been taught previously (be it from a programming, a journalism, or even a business background). The technological world is changing rapidly, and we must change with it. It's no longer good enough to fix old programs and applications; new programs have to be created, developed, and perfected. It's a large, scary business. But it's a rapid, fast-growing, and exciting one at the same time. Be prepared to learn techniques you never thought you would, and be sure to take on the good practices of those who have already developed so much of this technology.

**SPECTRUM:** ***What comes next? Is there any particular technology that you are interested in expanding into?***

**KELLY:** Mobile. We're already developing mobile applications for Prelude users.... but there's so much potential! Mobile is infiltrating warehouses, and it is being used as one of the main forms of communication between corporations. People are constantly on the move, and they need their business at their fingertips. Developing mobile tools that can connect them directly to necessary information is pertinent. And, heck, mobile technology is just plain fun to work with. IS

# The Rock and the Hard Place

## Parallel and Agile Development

### *Part 3: Must We Choose?*

BY SUSAN JOSYLN

*Note about the series: As we reach for the best way to deliver the highest quality of software and services we find ourselves considering parallel development and pondering the whole concept of agile. It may be that there is a sweet-spot between the two — and for each company that might be a slightly different spot. These articles will explore the perils of parallel development in part 1; the angst of agile in part 2; then how to pick and choose to find the right balance for your organization in part 3.*

Welcome to part three in which Pooh and his MultiValue friends gather the best honey from various trees to mix the perfect jar of software development lifecycle practices. To help in this endeavor we've compiled some nuggets of conventional wisdom, a few reference materials and some key terms to help in the search.

After examining the perils of parallel and the angst of agile, many of us may be looking for a third alternative. But we know from Fred Brooks' renowned paper *No Silver Bullet*[1] that, well, there is no silver bullet. There are no simple answers to our complex questions. Maybe we should just stick with what is tried and true. That may be a curmudgeonly, change-resistant attitude, but it is not entirely without merit. Obviously if we are still here developing software, we have found a good way to do it. We don't have to change our methods just for the sake of changing our methods. We are not trend-followers in the world of MultiValue! And unlike some of our IT brethren, we are not *chumps*.[2]

> *Not taking the time to refactor becomes a road block to all further attempts to streamline the lifecycle, modernize the interface or increase agility and productivity. There isn't a strong enough way to express the idea that we need to clean house a bit with some of our older programs.*

Yet, we are feeling pressure to accommodate parallel development and struggling with how to manage it. Our technology and our business is changing and growing more complex almost daily. No matter your intentions, you may find yourself staring down the neck of a bottle of "Agile." While some of the advertising may be tempting, the general consensus among practitioners is that trying to plug in some boilerplate set of practices, even with a cute sports-related or foreign sounding name, rarely brings sufficient returns to justify the effort and expense. And if we can't show those returns, we won't have the critical backing of upper management. The last thing we want to do

[1] Reproduced from: Frederick P. Brooks, The Mythical Man-Month, Anniversary edition with 4 new chapters. Originally printed for 1986 IFIP Tenth World Computing Conference. The paper "No Silver Bullet" likens software development to Werewolves. http://www.cs.nott.ac.uk/~cah/G51ISS/Documents/NoSilverBullet.html

[2] "Stevey's Blog Rants", a highly entertaining blog by Steve Yegge, a developer at Google . In the 09/27/06 article "Good Agile, Bad Agile", Steve suggests that agile methods are a meme gone viral and its adherents all case studies for Barnum's adage that there is a sucker born every minute.

is squander our credibility. Just like a fad diet where you will lose weight the first week, no matter the rules, then gain it all back in the second week, fad development practices are not sustainable. Yeah, you can hear it coming: It's a lifestyle change.

IT people on the ground have little time for theory and many have demanding business users that don't care how results are achieved; they simply expect them to be achieved, immediately. We've trained them, spoiled them that way. It is easier to push back when you are armed with solid understanding (that you can explain), believable metrics (that you can prove) and real world examples (that you can point to). We must be objective about current procedures and about the options. Get away from "What am I comfortable with?" and "What have we always done?" and move toward "What have we always done that is still working for us?" and "What have we been doing out of habit that may be costing us?" On one hand we don't want to be chumps, but we can't afford to put our heads in the sand, either.

You can't swing a cat in the world of software development without hitting a branded Agile Methodology. Of greater note is that you can't find a discussion about software lifecycle management these days that isn't about Agile in some form or another. Some of the terms and approaches that come from the Agile community have been so successful as to become commonplace. Those folks are definitely on to something. The trick is separating the baby from the bathwater — distilling what is useful and helpful for your organization from the overblown fluff or platform-specific dogma. We can help ourselves by lower-casing the word agile and eliminating the term "methodology" from our vocabulary. Japanese terms and sports analogies might best be avoided, too. Of course we need to be agile — fast moving, light on our feet, and flexible. But we don't have to adhere to some contrived set of practices in order to remain that way. We are ready to move into Chapter Three. Mark Kennally represents in his book "SDLC 3.0" that this is the rationalization of modern software engineering *methods* into a complex adaptive system of *practices*.

Remember when we were talking about parallel development and we looked at the "Cinderella" approach (go off in a corner with your part of the project and bring it back when it's done)? We looked at that idea from an agile perspective and the idea of interval merging came into view. It was an obvious way to connect the real requirement for parallel development with a faster-moving agile environment. Along that same line we could get away from the need for parallel development in many cases by breaking our change requests down into smaller deliverables and taking turns modifying a component instead of each taking our own copy off to our own little corner. This might be a staggered or overlapping style of agile implementation. Just as in Kindergarten, sometimes we share the toy, sometimes we take turns. (Author's note: The notion of staggered, alternating or overlapping projects is particularly appealing to this developer, who sometimes late at night likes to think of it as Staggering Agility.)

In reality, to move forward with smaller *projects* it may be necessary to have smaller *programs*. This means that those couple of multi-thousand line source programs that are central to the application and take heavy modification should be refactored. This is the most common argument in the MultiValue world for not being able to adapt agile practices and not being able to manage parallel development. Further to the argument is the notion that there is simply no time to go back and re-write those central programs into modularized functions. If one were to step back and count the hours wasted because of not doing it, the time could likely be justified. What stops us is not a cold, rational thought but a feeling - white-hot fear that comes from the belly. As in all things that are both scary and necessary, it is better to face it down, break it down. If you plan for it and take precautions, the very real risks can be mitigated. Not taking the time to refactor becomes a road block to all further attempts to streamline the lifecycle, modernize the interface or increase agility and productivity. There isn't a strong enough way to express the idea that we need to clean house a bit with some of our older programs. They are weighing us down like Granny's army boots.

Another bad habit that is holding us back is treating all projects as if they have the same levels of risk and complexity. This is happening on both sides of the coin — IT departments that still do things in a risky, uncontrolled way — even for projects that really ought to be done more carefully. And some IT departments that have gone the other way and created intense checks and balances and onerous procedures that have to be stepped through even for minor changes. Great returns can be realized from an early investment in identifying the programs, files, modules and projects that should be treated with more care and separating the ones that can be managed with a shortened or simplified life-cycle. Once you quantify them, mark them, document them, even the auditors will be happy with the shortened procedures.

After you do a little housekeeping you will find that you are better able to experiment with interval development. Intervals — time boxes — are a central thread of agile practices. Different brands define shorter or longer periods or other criteria for an interval. But in all cases we are admonished to move away from long, unwieldy development projects in a waterfall/cascading life-cycle and toward short term iterations with small, working deliverables.

This idea of timeboxes is so universally accepted that it has to be a great idea, right? Certainly it's good for 3rd party development companies who are quite happy to sign flexible contracts

*Continues on page 14*

## THE ROCK AND THE HARD PLACE: PARALLEL AND AGILE DEVELOPMENT

## PART 3: MUST WE CHOOSE?

*Continued from page 13*

allowing them to deliver "whatever" in "2 weeks" on a recurring basis until the client goes bankrupt. Okay, even if no one is really dumb enough to go along with that, there are other downsides to time-boxing. For one thing it is often an artificial, forced timeline. It has little to do with what is being delivered and nothing to do with who is delivering it. That is the idea, actually. That's what makes it work, when it works. But people have up days and down days. There is an internal rhythm to development that maybe can't be forced to march in step. The impedance mismatch can drive great engineers to mediocrity. But — now that you've identified areas of higher and lower risk, stratified your software repository — you can apply different timeboxes or lifecycles to different projects, to different programmers, to different periods of time.

This common theme of doing a little of an overall project, putting it out there, waiting for feedback, and then creating the next timeboxed deliverable has a lot of merit. It gets some resistance in our community because, let's face it, allowing this flexibility in requirements is the very definition of feature-creep. <little devil> It feels like it would lead to uncontrollable costs. For many organizations, the idea of letting go of the effort to block changes to requirements may feel like letting the dam break and being washed away. The idea lends itself to an all or nothing approach: The lifestyle change.

Let's get familiar with the history and terminology and dabble in a few of the variations. This can lead to a greater level of comfort with the idea, and it may point you to just the right book or set of practices to cherry pick whatever fits with your company culture.

To begin at the beginning, arguably the granddaddy of the agile movement is **Scrum**. A scrum is a way of restarting play in the sport of rugby after the ball has gone out of play. In 1986, a *holistic or rugby* approach was applied to product development (manufacturing), then later picked up mainly by Ken Schwaber in the 1990's who called it Scrum. When Schwaber collaborated with Mike Beedle on the book_*Agile Software Development with Scrum* in 2001, the name and the idea became firmly planted in our IT ideology. Scrum projects make progress in a series of sprints, which are timeboxed iterations no more than a month long. At the start of a sprint, team members commit to delivering some number of features that were listed on the project's scrum product backlog. At the end of the sprint, these features are *done*–they are coded, tested, and integrated into the evolving product or system. At the end of the sprint, a sprint review is conducted during which the team demonstrates the new functionality to the product owner and other interested stakeholders who provide feedback that could influence the next sprint. Over the last couple of years in particular, Scrum has garnered increasing popularity in the software community due to its simplicity, proven productivity, and ability to act as a wrapper for various engineering practices promoted by other agile methodologies.

The term **lean software development** originated in a book by the same name, written by Mary and Tom Poppendieck, which presents the traditional lean principles of manufacturing in a modified form. Lean Software Development focuses the team on delivering Value to the customer, and on the efficiency of the "Value Stream," the mechanisms that deliver that Value. Lean eliminates waste through such practices as selecting only the truly valuable features. Applying YAGNI (You Ain't Gonna Need It) for a system, prioritizing those selected, and delivering them in small batches. Like any agile approach, it emphasizes the speed and efficiency of development workflow, and relies on rapid and reliable feedback between programmers and customers. Lean uses the idea of work product being "pulled" via customer request.

**Kanban** is a process designed to help teams work together more effectively. A philosophical difference is that where Scrum places more emphasis on the project management practices, Kanban places its emphasis on business and value flow practices. If you love sticky-notes you may find yourself drawn to this set of practices.

The emphasis in **Extreme Programming (XP)** is in the technical practices. Some of these supporting practices have been controversial — and are not widely accepted. While no one can argue that having test plans is a good idea and that having the users involved in developing acceptance tests is obvious, many developers have complained that too much time is spent developing tests, taking away from the real task of developing software. Extreme Programming can be … well, extreme. Pair Programming came from this ideology and has taken a lot of criticism. Yeah, it is just what it sounds like — two people working at one station. The success of this approach may be dependent upon the personalities of the programmers.

Then we come to the **Crystal** family of methodologies. This is one of the most lightweight and adaptable approaches to software development. It predates the other agile methods and is more of a collection of meta-methodologies with the idea being that different projects and teams warrant different approaches. This is a people-centric approach and bears an underlying familiarity to MultiValue. Philosophically speaking. This Crystal family addresses the realization that each project may require a slightly tailored set of policies, practices, and processes in order to meet the project's unique characteristics. If you are looking for a place to dig deeper, Crystal might be a good starting point.

This is by no means a comprehensive list or complete history. But in all of the agile approaches the idea of allowing requirements to evolve over time and delivering bite-sized nuggets of working software mostly feels like a good idea. The differences in the various agile brands are often subtle but meaningful. It is quite reasonable to examine them and come up with your own workflow, your own criteria for an interval or a project, your own methodology for interfacing with users, empowering your team, for branching and merging and for delivering working software. Within your organization these components may vary from project to project, programmer to programmer, month to month.

It is, however, critical to create a workflow from start to finish so that there is guidance — a road map — to get software to the users. Probabilities for success are also greatly enhanced by having the right tools. Whether you consider a tool that imposes a methodology — but on the upside of that, supports it completely — or you settle on a few tools to help manage the rough spots, you want to make the procedure *do-able* as well *as know-able*.

The bottom line is that yes, you can manage parallel development in an agile way. Every organization has a sweet-spot, a balanced place between branching and agility. Working toward finding that balance will result in more satisfied users and developers. IS

**SUSAN JOSLYN is the President of SJ+ Systems Associates, Inc. and is the author of PRC?, a complete, integrated software development life-cycle management / IT Governance tool for U2. She has worked with U2 (nee Pick/Multivalue) and SB+ software the beginning (both hers and its) and has specialized in IT Governance, including quality, compliance and life-cycle productivity issues since the early 1990's.**

## LINUX SHARED MEMORY AND YOUR MULTIVALUE DATABASE

*Continued from page 9*

If you are still a skeptic about this being a real filesystem, try opening your favorite Linux shell editor and saving a text file there.

You might do the following:

```
nano -w /dev/shm/
     → thisiscool.txt
```

Type in some test text, hit Ctrl+X, and answer "yes" to save the file. Now enter:

```
ls -l /dev/shm
```

You should see your file there.

You might also be thinking, "Wait a minute, this looks an awful lot like a RAM disk. You mean this thing is taking up system RAM whether I use it or not?" The answer is no, /dev/shm is not a RAM disk in the traditional sense. It is shared memory, which means it looks like an old school RAM disk. It even smells like one. But it only uses RAM when you write something to it. When you delete something from it, it frees up the RAM.

Here is the best part. You can create an F-pointer in any of your accounts to /dev/shm and create items in your new shared memory file in your BASIC programs or your favorite Multivalue editor.

On OpenQM that F-pointer will look like this:

```
0001 F
0002 /dev/shm
```

On Universe the F-pointer is very similar:

```
0001 F
0002 /dev/shm
0003 D_VOC
```

Now before you go off creating F-pointers all over the place in your accounts, lets discuss a few caveats. First, there is security. You will notice the permissions on /dev/shm are pretty wide open, much like /tmp. So, one consideration is that you may want to mount /dev/shm with a little stricter permissions. Be careful in doing so, because you may have software on your Linux system that needs to write data there. Try doing this on a test system if you can before doing it on any live system. Be careful who has access to this new MultiValue file, and think about who can, and who cannot write to it. You don't want some average user dumping data there and filling up system RAM.

Second, understand that this should never be used to store any data that you want to keep for any real length of time. All data in /dev/shm is completely wiped out when you reboot the system. So don't start storing transaction history, sales data, demand history, and the like in /dev/shm, because on your next reboot, it will be gone in the twinkling of an eye.

The other nice thing about this method is it applies really well in the cloud where disk i/o is often very limited. Imagine if you could rent a $20 a month virtual Linux server in the cloud, install your MultiValue database, build your applications, and give users fast access to their data. Sure, a $20 a month server may not serve 1,000 concurrent users, but you might be surprised how far you can make it go with a few tweaks like this shared memory technique.

And last, I'm sure there are other uses for Linux shared memory in your MultiValue system besides using it as a transport layer for getting data to the web. Just give it some thought. I've met so many bright people in our community over the years, I bet with a little creativity and thought you folks can think of more. So what are you waiting for? Go get your hands dirty and silence the complaints of, "I can't get the data I need!" IS

**JOHN THOMPSON is currently employed at Bennett Auto Supply. He has been making technology (especially Multivalue) work in the business world for 9 years.**

# From the Press Room

## Datatel and SunGard Higher Education Combination Clears Regulatory Review

Hellman & Friedman, Datatel, SunGard Data Systems and SunGard Higher Education today announced that the proposed combination of Datatel and SunGard Higher Education has cleared Department of Justice review and the companies are preparing to close the transaction in the first quarter of 2012. As a result of this combination, over 3,000 employees will be dedicated exclusively to delivering solutions to education institutions in 40 countries.

"We have heard from many customers that they are eager to take advantage of the benefits that will be provided by this combination," said John Speer, Datatel President and CEO. "We expect to soon be in a position to deliver on this promise."

As noted when the proposed combination was announced on August 5, 2011, customers of the combined companies will benefit from:

Continued investments in current solutions: The combined companies plan to support the solutions of both companies going forward and to continue to invest significantly to enhance those solutions.

A broader portfolio of solutions: Customers will enjoy powerful new capabilities from the combined portfolio of products and services offered by Datatel and SunGard Higher Education.

Accelerated innovation: The combined companies will pool resources and talent to accelerate research and development with targeted investments focused on major challenges in the education community.

An expanded knowledge-sharing community: Institutions will experience increased collaboration, creativity and knowledge-sharing across an expanded global customer community of over 2,300 colleges, universities, foundations and state systems.

"This combination will help us achieve our goal of better serving customers and education communities globally," said Ron Lang, SunGard Higher Education CEO. "Our largely complementary products and services portfolios blend well, and our deep, shared commitment to our customers will continue to fuel the long-term relationships and innovative spirit we value so highly."

As described in the initial announcement of the proposed combination, affiliates of private equity firm Hellman & Friedman LLC will acquire the SunGard Higher Education businesses from SunGard Data Systems Inc. and combine the acquired businesses under a new holding company with Datatel, an existing Hellman & Friedman portfolio company. Closing of the proposed transactions continues to be subject to the satisfaction of customary closing conditions. ■

## Entrinsik Announces Release of Informer Web Reporting With Enhanced SQL Functionality

Entrinsik Inc. (www.entrinsik.com), developer of award-winning web-based reporting and analysis solutions for organizations across a variety of industries, recently announced the availability of powerful new SQL functionality for Entrinsik's Informer software. The new Informer Native SQL reporting brings an unprecedented level of scalability and flexibility to organizations using Informer against SQL databases.

Informer software provides real-time access to multiple databases including Oracle, SQL Server, MySQL, Access, DB2, Informix, U2 and others without complex ETL processes, data warehousing, or data cubing. From an intuitive drag and drop web interface, self-service reporting capabilities enable end users to drill down into data elements in real-time, sort and group, build interactive charts and perform ad-hoc analysis without IT support. Informer is implemented in hours and is easy to install, configure and administer, delivering quick ROI and a low total cost of ownership.

With the new functionality, Informer users can now enjoy the flexibility of creating an Informer report directly from a SQL query or stored procedure. Users who have existing SQL query statements now have the ability to immediately reuse these queries to create an Informer Report without the need to set up the metadata and mappings. Implementation and roll out is now faster than ever.

"Informer's new capabilities reflect our ongoing commitment to help organizations analyze data in real time, regardless of where it's stored, in order to make the best decisions possible," said Doug Leupen, President/CEO at Entrinsik. "Our developers continue to raise the bar with innovation that enables us to deliver a powerful yet easy-to-use reporting platform."

Watch a recorded webinar demonstrating the new Informer SQL functionality at http://bit.ly/InformerSQL.

Over 1000 organizations around the world use Entrinsik's software every day. For a personalized demonstration or a free trial, call us today at 888-703-0016 or visit www.entrinsik.com.

### Entrinsik, Inc.

Based in Raleigh, NC since 1984, Entrinsik develops, implements, and supports Information Management Solutions that enable organizations to maximize performance and improve bottom lines. Entrinsik's Informer Web Reporting is an innovative, award-winning reporting and analysis solution used by tens of thousands around the world. For a free trial, visit www.Entrinsik.com. 888-703-0016. ■

BY MIKE RUANE, REVELATION SOFTWARE

# OpenInsight and QM

## Such a Deal

OpenInsight, from Revelation Software, has been a true GUI MultiValue database and development tool since 1992, and has had tools for the Web and Browsers since 1996. We've worked on making our tools available to many of the MultiValue databases in the past. There is now a connector to Ladybridge System's QM database.

OpenQM, from Ladybridge Systems is the only MultiValue database available both as a fully supported commercial product and in open source form under the General Public License. As of OpenInsight release 9.3.1, there is a connector that will allow developers or users to use OpenInsight's tools, such as the GUI desktop, Banded Report Writer, or OpenInsight for the Web (O4W) with their QM database. It makes a very powerful combination and is surprisingly easy to set up and use. This article describes the theory behind how our connectors work, and how to use the QM Connector.

### Theory of the QM Connector: the MV BFS

OpenInsight has its own MultiValue database. OpenInsight runs on Windows and relies upon the operating systems to handle the physical reading from and writing to disk. OpenInsight's database and processing engine, called OpenEngine, relies upon a Base Filing System (BFS) to interface between the physical storage of the data and the calling Basic+ (our version of the BASIC programming language) program.

> *OpenInsight's QM Connector uses a BFS named MVBFS. This BFS relies on Ladybridge System's interface to the QM database. This interface is only available on QM versions 1.12-17 and higher.*

A BFS turns the Basic+ file requests into the logic required to access the physical data. The BFS makes the Basic+ program independent of filing systems by providing this translation layer.

There are as many BFSs as there are ways to store data. Each BFS can use a vastly different means of physically storing the data, so long as it accepts from and provides to a Basic+ program a conventional record, namely, a dynamic array.

OpenInsight also has a Modifying Filing System (MFS) that is a Basic+ routine that fits between Basic+ programs and the BFS. An MFS allows a developer to decide which of the operations made against a file (Read, Write, Select, etc) they wish to observe or modify in a Basic+ program.

OpenInsight's QM Connector uses a BFS named MVBFS. This BFS relies on Ladybridge System's interface to the QM database. This interface is only available on QM versions 1.12-17 and higher. Figure 1 shows a generic flowchart of OpenInsight and its BFSs.

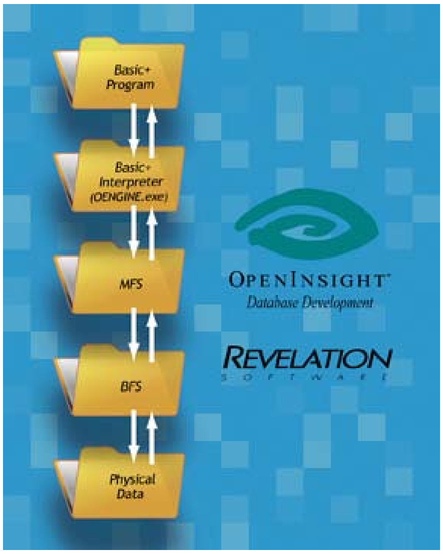
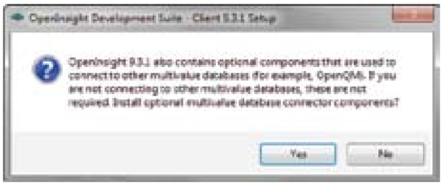


**Fig. 1 OpenInsight's Interface to Operating System Files**

### Requirements for Connecting to the QM Databases

In order to connect to QM, OpenInsight requires Ladybridge's OpenQM version 2.12-7 and above, OpenInsight version 9.3.1 and above, and the installation of the OpenInsight client on the workstation. OpenInsight ships with an executable named, oddly enough, ClientSetup.exe. This file, when run, offers to install various components used by various parts of OpenInsight such as reporting, .Net functionality, Data Encryption at Rest, and so on. You will need to run this client install on any client using OpenInsight. This install can be 'pushed' from the server, but for the purposes of this example, please run the ClientSetup.exe manually.

The Client Setup process will prompt you to install the necessary MultiValue database connector components. Choose "Yes" to install these components, as seen in figure 2.



OpenInsight Development Suite - Client 9.3.1 Setup

OpenInsight 9.3.1 also contains optional components that are used to connect to other multivalue databases (for example, OpenQM). If you are not connecting to other multivalue databases, these are not required. Install optional multivalue database connector components?

Yes No

**Fig. 2**

## Connecting to the QM Databases

The first step in using OpenInsight's QM Connector is to establish a connection to the QM data source. This can be found in OpenInsight by opening the Database Manager and choosing Utilities-Other MV Tables-OpenQM-QM Connection Wizard. (Fig. 3)

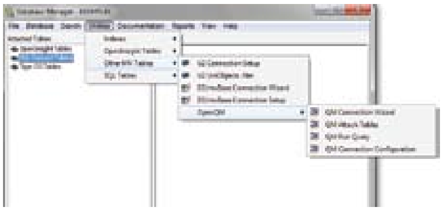
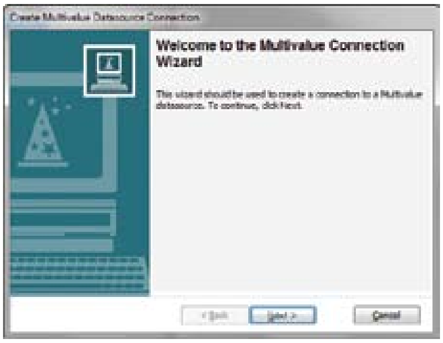


**Fig. 3 Choosing the QM Connection Wizard from the Database Manager**

This will call up the MultiValue Connection Wizard, as seen in figure 4.



Create Multivalue Datasource Connection

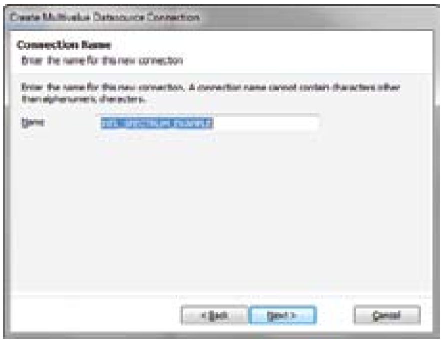
Welcome to the Multivalue Connection Wizard

This wizard should be used to create a connection to a Multivalue datasource. To continue, click Next.

< Back Next > Cancel

**Fig. 4 The MultiValue Connection Wizard**

First off, we need to give this connection a name. This name will be used by OpenInsight as an identifier for the connection, and will allow OpenInsight to 'Attach', or create a logical tie, to the files/tables that will be used at that location.(Fig. 5)



Create Multivalue Datasource Connection

Connection Name

Enter the name for this new connection

Enter the name for this new connection. A connection name cannot contain characters other than alphanumeric characters.

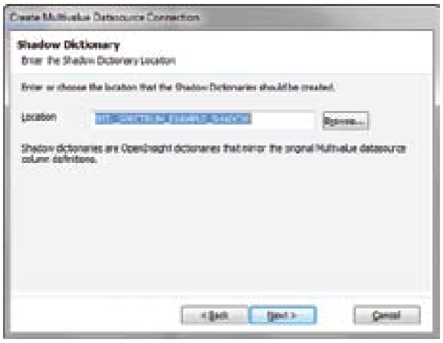
Name: INTL SPECTRUM EXAMPLE

< Back Next > Cancel

**Fig. 5 The Connection Name Panel**

Clicking on the 'Next' button brings up a panel that prompts for the location of the Shadow Dictionary location. Shadow Dictionaries are OpenInsight versions of the column definitions that define the OpenQM file. OpenInsight does not want to affect the column definitions on the OpenQM data source, so a local copy is made.  There are a number of advantages to this, including formatting for OpenInsight reports.

The Wizard offers a default folder location where the shadow dictionaries will be created.  If a full path is not specified a directory will be created within your OpenInsight folder. We will accept this default, and then click on the 'Next' button. (fig. 6)



Create Multivalue Datasource Connection

Shadow Dictionary

Enter the Shadow Dictionary Location

Enter or choose the location that the Shadow Dictionaries should be created.

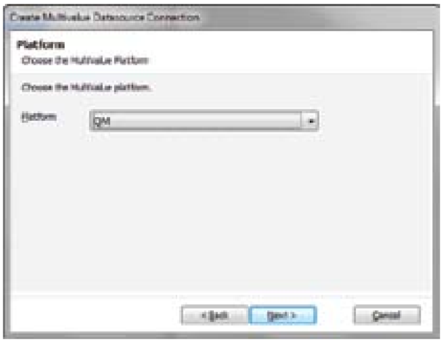
Location: INTL_SPECTRUM_EXAMPLE_SHADOW Browse...

Shadow dictionaries are OpenInsight dictionaries that mirror the original Multivalue datasource column definitions.

< Back Next > Cancel

**Fig. 6 The Shadow Dictionary Panel**

The Platform panel is where the user will choose the flavor of the MultiValue system they are going to connect to. Currently, OpenInsight has separate connectors for U2, D3, mvBase and QM. Our connector technology has advanced to the point that we are able to use a common wizard to connect to all the different MultiValue flavors. While this Wizard will eventually be modified to be the wizard for all MultiValue data sources, it is currently set up only for QM. Click on the Next button to proceed. (fig. 7)



Create Multivalue Datasource Connection

Platform

Choose the MultiValue Platform

Choose the MultiValue platform.

Platform: QM

< Back Next > Cancel

**Fig. 7 The Platform Panel**

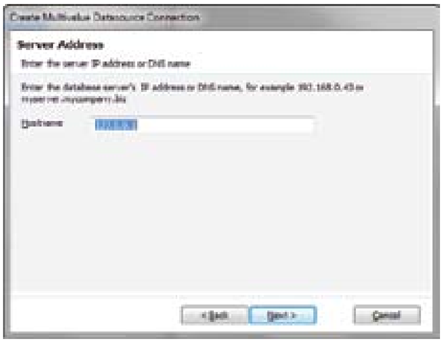
The next panel displayed is where the IP address of the OpenQM server is entered.

*Continues on page 20*

## OPENINSIGHT AND QM – SUCH A DEAL

*Continued from page 19*

The IP Address lets OpenInsight know the address of the server, either on the local machine, the local network, or the World Wide Web. The OpenQM server can be Widows, Unix, or Linux based. The wizard defaults the IP Address to 127.0.0.1, which is the local host machine. You should either change it to the address given to you by the System Administrator, or accept the default. Click on the Next button to proceed to the Server Port Panel. (fig. 8)


Create Multivalue Datasource Connection

**Server Address**

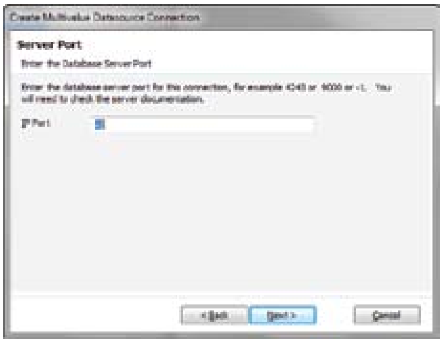
Enter the server IP address or DNS name

Enter the database server's IP address or DNS name, for example 192.168.0.43 or myserver.mycompany.biz

Hostname: 127.0.0.1

**Fig. 8 The Server Address Panel**

Clicking on the 'Next' button brings up the Server Port panel. The Server Port number is the communication port number on the server that OpenInsight will connect through. OpenQM version 2.12-17 supports port number -1, so that is what OpenInsight defaults to. An entry of -1 will find the correct OpenQM port automatically. You can either accept the default, or change it to the Port Number given to you by the System Administrator. (fig. 9)


Create Multivalue Datasource Connection

**Server Port**

Enter the Database Server Port

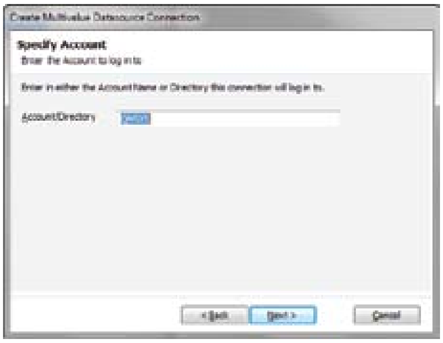
Enter the database server port for this connection, for example 4243 or 9000 or -1. You will need to check the server documentation.

IP Port: -1

**Fig. 9 The Server Port Panel**

Clicking on the 'Next' button brings up the Account/Directory panel. The Account/Directory Panel is where the user enters the name of the account where the OpenQM files are stored, or the directory that holds the QM files the user wants to work with.

For the purposes of this exercise we will use the QMSYS account. Enter that name and click on the 'Next' button. (fig. 10)


Create Multivalue Datasource Connection

**Specify Account**

Enter the Account to log in to

Enter in either the Account Name or Directory this connection will log in to.

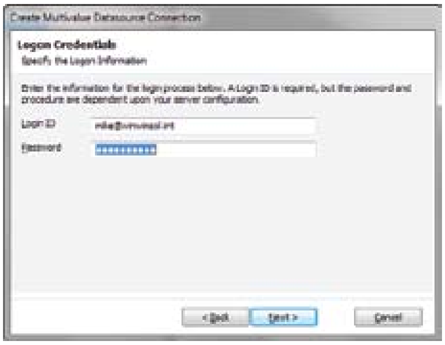
Account/Directory: QMSYS

**Fig. 10 Specify the Account to connect to**

Clicking on the Next button will bring up the Logon Credentials form. The Logon Credentials screen is where the user enters the information needed for OpenInsight to connect to the OpenQM server.

The Logon ID is a username accepted on the server. The Password is the password required by the Logon ID.

For this example, I'll use my username and password, and then click on the Next button. (fig. 11)


Create Multivalue Datasource Connection

**Logon Credentials**

Specify the Logon Information

Enter the information for the login process below. A Login ID is required, but the password and procedure are dependent upon your server configuration.

Login ID: mike@[illegible]

Password: ●●●●●●●●●●

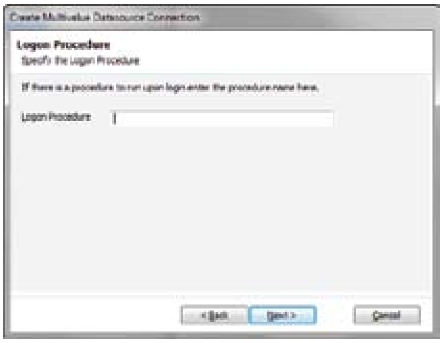
**Fig. 11 The Logon Credentials Panel**

The next panel asks information about the Logon Procedure. The Logon Procedure is the name of a procedure that will run on the QM server when OpenInsight connects using the Logon ID specified.

Here's an important point: If your QM LOGIN procedure has prompts or inputs, it is necessary to insert a line of code such as:

```
IF @TTY = 'vbsrvr'
     → THEN STOP.
```

This line allows the paragraph to terminate correctly for the QM Connector users. This stops things like menus, etc. from trying to appear and corrupting the client/server communication. For this example, we will not use a login procedure. (fig. 12)


Create Multivalue Datasource Connection

**Logon Procedure**

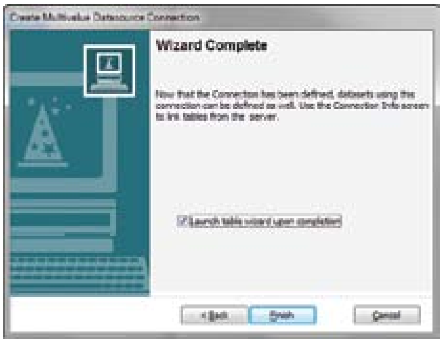
Specify the Logon Procedure

If there is a procedure to run upon login enter the procedure name here.

Logon Procedure:

**Fig. 12 The Login Procedure Panel**

Clicking on the next button brings up the Wizard completion panel. This panel has the usual 'Finish' button, but there is also an additional checkbox labeled 'Launch Table Wizard upon completion'. If this box is checked when the Finish button is clicked, a secondary wizard is launched where the user can choose which files to make use of in the QM application that they are connecting to. For the purposes of this example, let us presume the checkbox is checked, as seen in Figure 13, and we'll click on the Finish Button.


Create Multivalue Datasource Connection

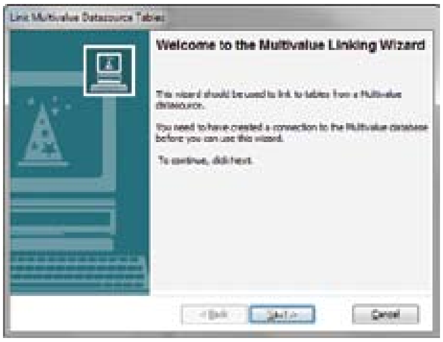
Wizard Complete

Now that the Connection has been defined, datasets using the connection can be defined as well. Use the Connection Info screen to link tables from the server.

☑ Launch table wizard upon completion

Fig. 13 The Wizard Completion Panel

Clicking on the finish button will update OpenInsight's file definitions, and a couple of messages may flash across the screen. After that, the MultiValue Linking Wizard will appear, as seen in Figure 14. Click on the Next button to begin.


Link Multivalue Datasource Tables

Welcome to the Multivalue Linking Wizard

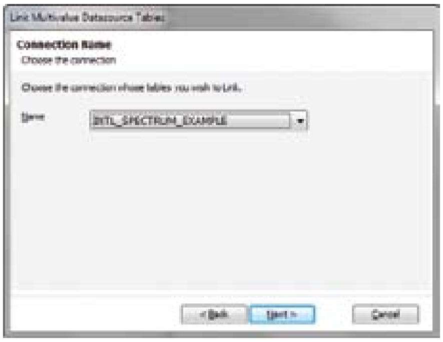
This wizard should be used to link to tables from a Multivalue datasource.

You need to have created a connection to the Multivalue database before you can use this wizard.

To continue, click Next.

Fig. 14 The MultiValue Linking Wizard

Once the next button has been clicked, the Connection Name panel appears. This panel contains a list of all the connections defined for the application. Choose the connection that corresponds to the QM application whose tables you want to connect to. For the purposes of this example, we'll choose the INTL_SPECTRUM_EXAMPLE connection, and click on the Next button, as seen in Figure 15.


Link Multivalue Datasource Tables

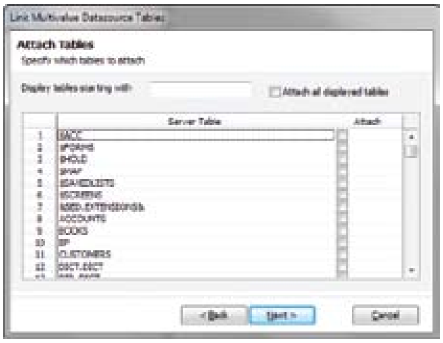
Connection Name
Choose the connection

Choose the connection whose tables you wish to Link.

Name: INTL_SPECTRUM_EXAMPLE

Fig. 15 The Connection Name Panel

Clicking on the Next button will call up the Attach Tables panel. This panel shows a list of all the tables (or files if you prefer) in the QM application that is being attached, as seen in Figure 16.


Link Multivalue Datasource Tables

Attach Tables
Specify which tables to attach

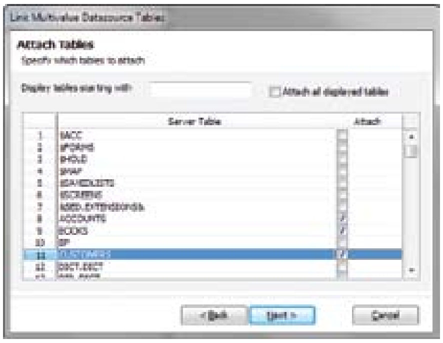
Display tables starting with:

□ Attach all displayed tables

| | Server Table | Attach |
|---|---|---|
| 1 | $ACC | □ |
| 2 | $FORMS | □ |
| 3 | $HOLD | □ |
| 4 | $MAP | □ |
| 5 | $SAVEDLISTS | □ |
| 6 | $SCREENS | □ |
| 7 | $SED.EXTENSIONS& | □ |
| 8 | ACCOUNTS | □ |
| 9 | BOOKS | □ |
| 10 | BP | □ |
| 11 | CUSTOMERS | □ |
| 12 | DICT.DICT | □ |

Fig. 16 The tables in the QM application

The user can then decide on which tables they want to connect at that time by clicking on the check boxes in the grid, as seen in Figure 17. After the tables have been chosen, click on the Next button. That will bring up the Wizard completion page, where the use should click on the Finish button.


Link Multivalue Datasource Tables

Attach Tables
Specify which tables to attach

Display tables starting with:

□ Attach all displayed tables

| | Server Table | Attach |
|---|---|---|
| 1 | $ACC | □ |
| 2 | $FORMS | □ |
| 3 | $HOLD | □ |
| 4 | $MAP | □ |
| 5 | $SAVEDLISTS | □ |
| 6 | $SCREENS | □ |
| 7 | $SED.EXTENSIONS& | □ |
| 8 | ACCOUNTS | ☑ |
| 9 | BOOKS | ☑ |
| 10 | BP | □ |
| 11 | CUSTOMERS | ☑ |
| 12 | DICT.DICT | □ |

Fig. 17 Tables selected from the QM application

## Testing the Connection

A very quick way to ascertain as to whether or not the connection is working is to use OpenInsight's Table Browser. From the Database Manager, look at the Attached table in the left-hand panel. Click on the MV tables, expand the list of connections, and expand the connection you just made, in our case the connection named INTL_SPECTRUM_EXAMPLE. Once the list of tables has expanded right-mouse-click on one of the tables. In our case, we'll choose the CUSTOMERS table, and choose the Table Browser, as seen in figure 18.

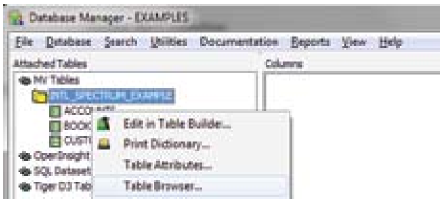


Fig. 18 Choosing the Table Browser from the Context Sensitive Menu

Once the Table Browser has appeared, and the load button has been clicked, the Table Browser will be populated with data from the table, as seen in Figure 19.

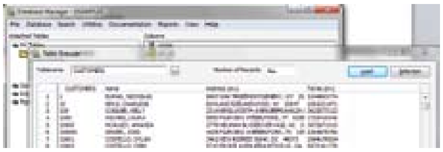


Fig. 19

## And now...

Is that it? No, of course not, but now you have the entire OpenInsight toolset available for your use. Perhaps your green screen application is not dead; perhaps your green screen application is the server for an OpenInsight GUI client, or better yet, a browser-based application that will run on any desktop or mobile device with a browser. Pair these two excellent products to maximize your existing investment in MultiValue tools, maximize your MultiValue knowledge, and maximize your sales and consulting time. IS

BY CHARLES BAROUCH

# Why Programmers Should Plan

As programmers we all have had the instinctive urge to ditch the planning phase of a project and jump in with our metaphoric guns blazing. Personally, I know I've given in to that urge a time or two.

However, planning has some meaningful benefits. One of the biggest has nothing to do with the project; it has to do with future employment.You see, if you keep the coding separate from the design,you can often apply your design skills to projects which will ultimately be written in other languages. This yields a few different benefits:

1. You are promotable from coder to analyst.
2. You can make system-neutral arguments which can help justify keeping and expanding the MultiValue systems you support.
3. It is easier for you to survive in a company which moves away from MultiValue, increasing your options.
4. It lets you learn new languages and incorporate them into your tools, making you a more valuable employee or consultant.

> *I never do a project on the assumption that I will never have to do anything similar.*

Think of it the way you think about tuning a car. You can learn how to tune a 1979 Impala, or you can learn how to tune pre-fuel injection cars in general. Learning the Impala, you might pick up some tricks which are specific to that car, making you better with that model.However, learning tuning at the broader level means you can do a good job in more situations,even if you can't do the absolutely best job in the one specific case.

Nothing is stopping us from learning both.

### Planning to Fail

One of the exciting aspects of planning is that it helps you fail faster. Given a choice between hitting a dead end six months into a project or six days into a project, I vote for the faster one.The sooner,and therefore cheaper, I find the roadblocks, the better my client or employer can make decisions that allow the project to restart or readjust. I've worked on projects where planning has allowed me to convince the boss to hire more staff or redeploy the folks we already had. I watched someone lose a job once because they didn't raise their hand soon enough. By the time management knew that things were at a standstill, the investment of time and money was so high that someone was getting fired.

So, I don't mind failing early. It is much easier to adjust the plan and try again than to adjust a million lines of code. Planning, unlike coding, is a journey you might be able to share with other, less technical people. I've had any number of visionaries, who despite being code-illiterate, understood so much more about the business than I did. Coding as I went would have cut me off from their participation. Sometimes a failure isn't considered a failure if it is discovered communally and resolved by the group.

### Show Your Work

In math class, you could often get partial credit for a wrong answer if you showed your work. More importantly, the teacher could show you where you went wrong so that you had a chance to do better next time. I never do a project on the assumption that I will never have to do anything similar. Every project is prep for the next project. So, the better we set up the experience for learning, the less pain we feel on the next project.

When we plan, we show our work to others, including them in the process. We increase the number of voices—for good and bad—and we spread the understanding of why we do what we do. This brings me to the story of a chemist friend of mine. She called me one night in an attempt to make sense of her boss's recent hostility.When she started the job, they got along famously. He was her mentor. Recently, she had begun working more independently, and he had started shooting down all of her projects. She didn't think he could be jealous, she was still well below his level, but she was at a loss for a better reason.

So, I asked her if she had come to him frequently with questions in the beginning and she says, "Of course." Then I asked her if she thought that the questions gave him some insight into her thought process, and she said, "Absolutely." So then I asked if she was now presenting conclusions without explaining her thought process and she said,"That's it!"

The next day she presented a new project and included a section on her thought process, her dead-ends, and her eventual conclusions. He approved the project. Showing your work is very useful.

### Leap of Faith

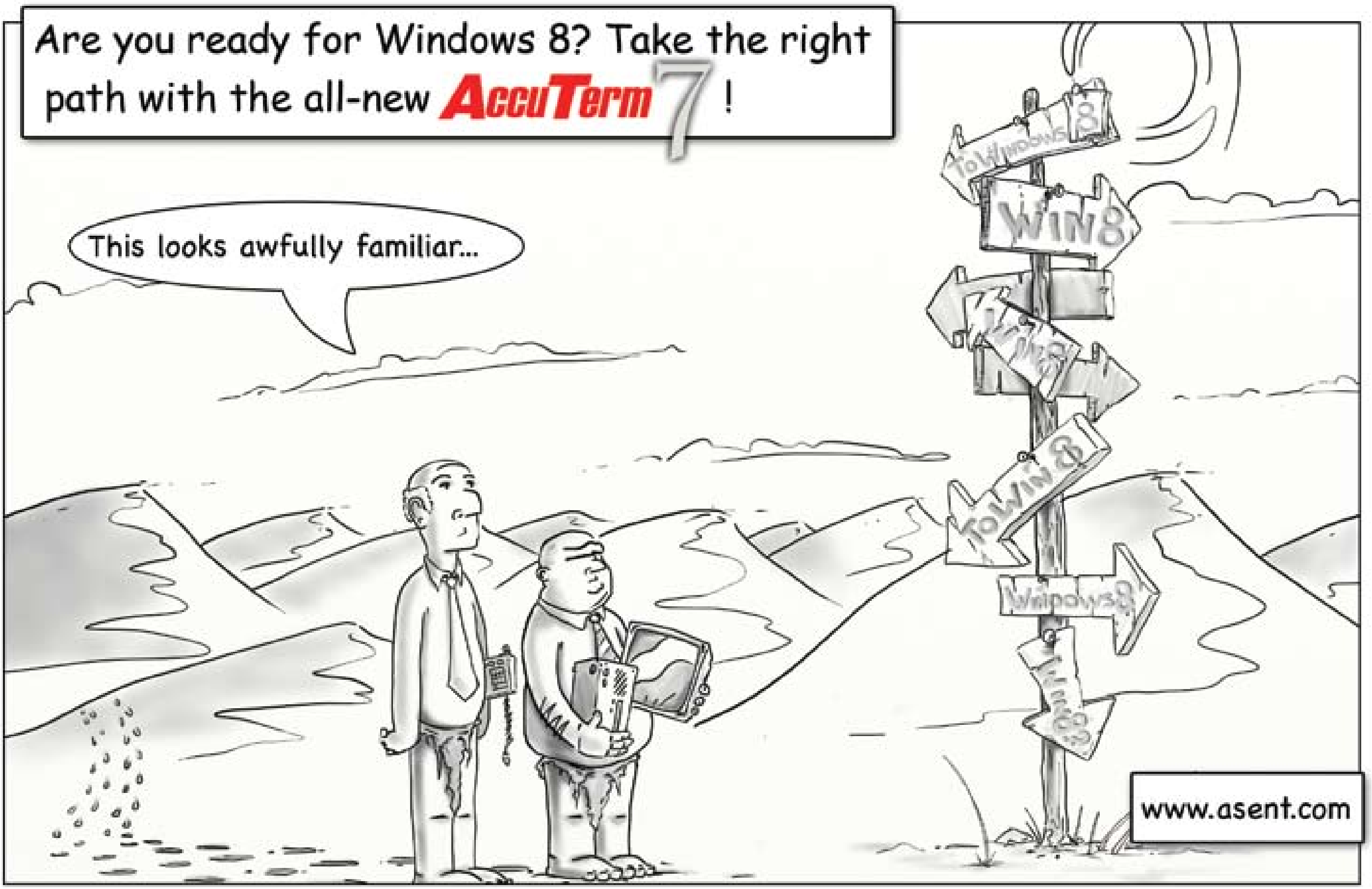
Let's assume that you believe that planning has a sufficient value. The next step is to understand how to maximize that value. When I have the option of setting the pace, which is certainly not all of the time, here's my preferred method:

1. Meet
2. Build a Straw Man
3. Build a Plan
4. Get Buy-in

Meetings, to quote one former coworker, are often a group of people sitting around a table, trying to contact the living. What we are looking for here is some healthy exchange. Generally, I try to get one of the people who need the project talking and then I ask a few wide-open questions. The more general the question, the better the chance of getting a useful answer. "What's the biggest issue we need to solve?" vs. "Do you want that as a report or an inquiry?"

Once they are talking, get the other people engaged. If the project serves Accounting and Sales, make sure you get both sides talking. Otherwise you end up with half a project specification. Hopefully, the key people will start interacting and refining the idea.

*Continues on page 24*


Are you ready for Windows 8? Take the right path with the all-new AccuTerm 7 !
This looks awfully familiar...
To Windows 8
WIN8
To Win 8
Windows8
Win8?
www.asent.com

## WHY PROGRAMMERS SHOULD PLAN

*Continued from page 23*

The Straw Man is a term used for a military practice dummy. It is something that can be cobbled together quickly, ripped to pieces, and remade as needed. My Straw Man is the meeting notes. Publish the notes in summary to all parties and tell them that you aren't positive you have it all right in your notes, i.e., ask them to rip it apart. Most people can't write a top flight IT project specification. Almost anyone can point at your notes and say, "Line three is wrong.We never sell single units."

Once the Straw Man has been savaged— perhaps in more than one goround— you can build your plan. The neat thing is, you have done very little work at this point.You sent one e-mail after sitting in one meeting. Instead of taking the entire project—win or lose —on to your shoulders, you have a group doing the bulk of the functional specification for you.You have people starting to feel connected to the project. You have some people besides you who see a success as their success. This will make them more helpful when we need to test or revise later. After all, it isn't your project, it's everyone's project.

When you build a house, you consult non-builders on things like the number of stories. Those answers imply technical answers about how solid the foundation has to be, even though you never asked about the foundation. In all technical fields, certain A information implies certain B information. So, while my customer didn't tell me to program in Delphi instead of PHP, I know the strengths of each language I use and I can make a best choice analysis from the facts I have been provided.

Likewise, in a MultiValue project, I should be able to select the right tools for the job at hand. That might mean introducing a 4GL, a reporting tool, or other existing technologies. It might mean leveraging the tools which the company already owns. It might mean redesigning database files. Moving from the business level specification to the technical specification requires you to have more than one skill, so that the choices are derived from the best course of action instead from the limits of the team.

Let's use a practical example and see where it takes us. Walt has a client with a billing system that was designed for a small medical practice. The client has a small medical practice, but they've elected to start a clinic and they don't want to abandon the system they have; they want to expand on it.

So, following the practices we just outlined, Walt meets with all the interested parties, sends out his straw men, and reaches a consensus.

Since the practice uses a 4GL already, he's going to do the project with those tools. Since the 4GL uses the Web as an interface, Walt uses his business plan to convince the client to hire a web designer to make the pages look right and flow smoothly. Now, he has to make the leap from business plan to technical plan, and layout the project.

### Architect-Ive

Walt starts his technical plan by reviewing the existing database architecture. He wants to make the project work smoothly with the existing billing system, so he needs to understand the relationship between the fields and files. This will allow him to selectively add fields and possibly files to the layout without duplication or conflict.

### Pseudo-Coder

Next, Walt needs to rough draft the programs and flow. Wisely, he throws out every hint or suggestion that uses a flow chart. What? Yes, Walt knows that flowcharts are useful in explaining technical matters to non-techs. He may actually design the charts — or some of the charts — after the project is complete. He won't use them in the design process because flowcharts have a fatal flaw. They take up more room than almost any other way of describing a process. As a result, you have to either pin them to a wall and walk back and forth, or page through them, or print them so small as to make them barely readable. None of these options make them easy to keep in your head.

A much better alternative, the one Walt chooses, is to pseudo-code. He writes a

```
SUBROUTINE Billing.Correction(Phase, Params, HTML, Status)
* by Walt Aminit
* on 04/01/08
* ***********
* This program is designed to allow
* for adjustments up until the bill
* is printed.
* ***********
* Initialize
* If Phase=new:
* --Paint Screen (Web Page)
* If Phase=check:
* --Evaluate Params
* --Send back errors in Status
* --Send back errors in HTML
* --If no errors, send user to next page
RETURN ;* Logical End of Program
*
* Start of Subroutines
*
END ;* Physical End of Program
```

**Fig. 1**

series of programs that look like figure 1.

This gives him a very compact way of looking at the program flow. Unlike a flowchart, it fits on one page. Unlike a flowchart, he can stick code between the lines, which turns these comments into internal documentation. Unlike a flowchart, if he strays from the design, the comments are right there to remind him that he is changing things, so he can decide to keep to the original plan, or have a clue that he needs to amend the downstream code to reflect that he has chosen to intentionally stray from the design.

As he moves forward, his code moves with him, as you can see in figure 2. He adds some additional pseudo-code as he goes, but the only real code he adds at this point is branching code (GOSUB, CASE, IF, etc.). He does this because getting the skeleton right is easier before the working code is installed.

Oddly, many non-programmers can read a program that is just comments and branches. So, pseudo-code can be superior to the flowchart in this way as well.

The middle ground between the pseudo-code and the business spec is called the “use case.” It is a method for making software more robust by setting up a series of “what if this happens” write ups:

- What if the customer wants to split the bill between two credit cards?
- What if the customer wants to pay in installments?
- What if the check bounces?

When a business person thinks in use cases, reading CASE statements becomes almost natural to them.

### Rad Catcher

Since the client is using a 4GL, Walt might have another starting point besides pseudo-code. He could consider RAD (Rapid Application Development). While RAD is possible without a 4GL or an object-oriented language, it is much easier with one of these tools. One of the biggest problems in completing software projects lies in designing interfaces which make sense to the people who will be using the project. The idea behind RAD is that designing the screens rapidly can save huge amounts of time and effort.

Where pseudo-code is primarily about the process, RAD is primarily about the

Continues on page 27

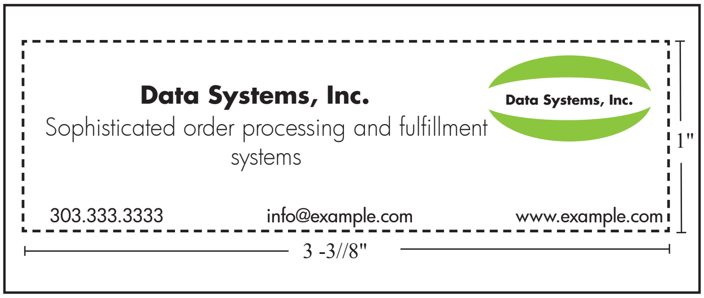
```
SUBROUTINE Billing.Correction(Phase, Params, HTML, Status)
* by Walt Aminit
* on 04/01/08
* ***********
* This program is designed to allow
* for adjustments up until the bill
* is printed.
* ***********
* Initialize
GOSUB INIT
*
BEGIN CASE
    CASE Phase = “N”
        * If Phase=new:
        * --Paint Screen (Web Page)
    CASE Phase = “C”
        * If Phase=check:
        * --Evaluate Params
        * --Send back errors in Status
        * --Send back errors in HTML
        * --If no errors, send user to next page
    CASE 1
        * Error in Phase, only “N”ew and “C”heck
        * Are supported
END CASE
RETURN ;* Logical End of Program
*
* Start of Subroutines
*
INIT:
* TBD - Add Initialization Code
RETURN
*
END ;* Physical End of Program
```

**Fig. 2**

# MARKETPLACE

| ADVERTISER INDEX | | | |
|---|---|---|---|
| AccuSoft Enterprises | 23 | Ladybridge Systems, Ltd. | 30 |
| Execu-Sys, Ltd. | 19 | Natec Systems | 27 |
| HDWP | 11 | PICK Programmer's Shop | 11 |
| InterSystems, Inc. | 2 | Revelation Software | 4 |
| Kore Technologies | 9 | SJ+ Systems Associates | 7 |

## HIGH IMPACT 4-COLOR DISPLAY AD



**Data Systems, Inc.**
Sophisticated order processing and fulfillment systems

303.333.3333 info@example.com www.example.com

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

## WHY PROGRAMMERS SHOULD PLAN

*Continued from page 25*

visual elements. So, Walt can start by putting together a quick entry screen and treat it just like he treated his other straw men.

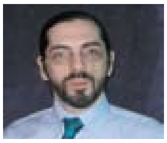
"Sally, have a look at this page for the billing entry. Let me know what options you need that I've left out."

This allows us to get the comments, complaints, and compliments before the code is finished. Personally, I'd be willing to jump through a lot of hoops to avoid having to do fundamental redesign on projects that are already having birthdays. RAD doesn't require a lot of hoops, but it does dramatically reduce the amount of late-phase redesign.

RAD extends the amount of time during which the business professionals can actively contribute to the design and implementation of the project. Most 4GLs allow you to create working screens (with little or no validation) in a few minutes. Walt elects to do both RAD and pseudo-code, so that he develops the engine at the same time as he builds up the screens. Since both techniques allow you to make mistakes and redesign quickly, he isn't trapped by any of the decisions he makes early on and can learn from his peers on the non-technical side.

- When Walt is done building a usable version of the software, he has achieved several important goals:
- Walt's client knows why he has made the choices he has.
- The client has an emotional stake in the success of the project.
- The project has a high chance of meeting the real needs.
- The skills he has applied make him more valuable on future projects.
- Walt now has friends and allies at the client. People trust that he can listen and deliver.
- Because he has included them in the design cycle, the perceived time between the end of the spec and the release of working code is shorter.

If the next project uses a different 4GL, a different database, or serves a different business need, Walt is still a strong candidate because he can do the analytics even in a situation where he can't do the coding. It seems that project planning not only increases the chance of success but it also improves your reputation. IS



**CHARLES BAROUCH** is the CTO of HDWP, Inc. He can be contacted at www.hdwp.com

Tech Tip

BY MARK JOHNSON

# Spooler Management (D3, Pick, R80 specific)

One of the most powerful commands is SP-ASSIGN for assigning the output to be printed (or managed) either currently or at a later time.

While SETPTR exists for many other MultiValue platforms, I will focus on SP-ASSIGN and its derivatives for this article. Those familiar with SETPTR can easily convert my suggestions to use on those platforms.

Most of the time the printers are created in a proc stored in the DM (or SYSPROG) account accessible by the USER-COLDSTART process. Thus, when the system is booted, the printers are created once and remain available.

I have created a regular data file called PRINTERS and a program called CREATE.PRINTERS that will replace the details within that PROC. Basically the PRINTERS file's primary key would be F0, F1, F2 and field 1 would be the network printer name, field 2 would be the serial port number, and field 3 would be the SKIP parameter.

Thus, the CREATE.PRINTERS program would SELECT the PRINTERS file and create the appropriate TCL commands for each printer with SP-KILL then DEV-MAKE

then STARTPTR. Plus, this program could be modified to create a single printer if that printer were provided in the TCLREAD string.

As new printers are added to the system, a simple new entry in the PRINTERS file would be added and this program is run again. If anything is changed to existing printers, again the data is changed in the PRINTERS file and this program is run. One-stop shopping.

Additional fields in the PRINTERS file could be added for your local use. For example, Name, Location, Paper type and TERM parameters. As your printer environment gets more complex, you can get a handle on the stable of printers available.

The Name field would be HP-2040 or Printronix P600. The Location would be Accounting or NY-Sales or CA-Warehouse. The Paper type would be Laser, Greenbar, Checks, Invoices, Orders or other paper types. The TERM parameters are obvious.

One client of mine has 41 printers covering a large array of these fields. They have 20 laser printers, 3 Printronix greenbar printers, 9 Okidata dot matrix printers, 2 MICR laser printers for Checks, 5 barcode/label printers and 5 PC-attached inkjet printers all available as application printers in their 4 physical locations.

This entire application only has one occurrence of the command SP-ASSIGN. It is buried within a program called SP.ASSIGN (dot versus dash) that will intelligently process the request way beyond the original design of SP-ASSIGN. All the procs and programs that want to print on their appropriate printer all use the SP.ASSIGN (dot) command.

Participating with this PRINTERS file is an extension of the USERS file such that for every one user, there are fields that indicate their default greenbar printer, default laser printer, Order printer, Invoice printer etc. A maintenance program will assign the default printer for that USER for the specific printer needed.

In addition to having one specific printer assigned to a specific user for that function, there are the choices "D" for download, "HS" for printing later, "AL" for asking which laser printer, "AG" for asking for which greenbar printer etc.

So instead of the hard-coded `SP-ASSIGN F5` inside a proc, the command `SP.ASSIGN GREENBAR` would exist and the `SP.ASSIGN` program would interpret `GREENBAR`, get the user's info and either hard-assign the printer designated or, using English, prepare a list of Greenbar printers to choose from.

I designed and implemented this concept years ago for my clients. A recent request from those clients was to somehow 'download' any report that would normally be printed.

No problem. Having replaced `SP-ASSIGN` with `SP.ASSIGN` allowed me to allow the letter "D" for any of the printer assignments. The `SP.ASSIGN` program will always fabricate the appropriate `SP-ASSIGN` command with its parameters. For downloads, it uses the HS option.

The whole print job goes into the `PEQS` file, and that entry number is available with `SYSTEM(20)`. Thus you would read the output, download appropriately, and usually delete it.

A more recent request was to put the report into the Microsoft Clipboard. Again, no problem. I simply allowed the answer "CL" to be in the user's printer field and `SP.ASSIGN` would process it the same as a download but put it into the clipboard. Consult your specific terminal emulator for both download and clipboard functions.

Printer assignment has always been tricky as our environments get more complicated. The limitations of having all those procs condition the `SP-ASSIGN` command caused me to consolidate this madness into those procs 'requesting' a printer using the `SP.ASSIGN` program to build the proper `SP-ASSIGN` command. Plus, it is smart enough to even process `SP.ASSIGN 5  F19  HS` as if it were `SP-ASSIGN`.

I have run into many long-term programmers who are slightly intimidated with all the spooler commands. Going through the exercise of consolidating `SP-ASSIGN` towards `SP.ASSIGN` will give you a better understanding of this beast called the spooler.

If you need to send `PCL` commands prior to printing for specific printers, you can manage that here on a large-scale basis instead of within each program. This is used for printers where their defaults cannot be modified.

Finally, if you ever port your app from D3 to the SETPTR family of MultiValue platforms, you would not have to change all the `SP.ASSIGN` references. Just change what the single `SP.ASSIGN` program does, SETPTR-wise.

A program demonstrating the use of these techniques can be found at intl-spectrum/s1050 **IS**

*Do you have a Tech Tip to share? E-mail it to editor@intl-spectrum.com*

## JSON: BORN FOR MULTIVALUE

*Continued from page 7*

price) with corresponding values for each. As associations get larger and more complex — perhaps even using subvalues — JSON objects like this can be a very useful way to represent a complex data structure.

Okay, so now that you know how to build JSON, you're probably wondering… what's the big deal, especially considering that few — if any — MultiValue platforms natively support it?

The big deal is that while the MultiValue community at large may be slow at adopting JSON, the rest of the world is not. JSON is well supported in dozens of contemporary languages, thus opening a door for us to trade our MultiValue information with an ever-widening audience. Getting information out of MultiValue in JSON format is easy; deep multidimensional JSON structures can be constructed in MultiValue using nothing more complicated than simple string concatenation. Getting information INTO our MultiValue systems using JSON is a bit more involved, but certainly a far cry from impossible. Furthermore, as vendors offer increased support for JSON, using JSON to import information into our MultiValue systems will only get easier.

I've contended for years that for MultiValue to continue to grow and thrive we have to get our technology more involved in the global technology community as well as incorporate global technology into our own. Stated another way, we need to better support people and programs connecting to our systems, and we need to be actively working to connect to theirs.

The future will not be driven by those maintaining islands of superior technology but rather by those who encourage cooperation amongst diverse information producers and consumers, regardless of the technology. It is with this in mind that JSON provides a brilliant foundation for a bright, bright, cooperative future. **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.**

## CLIF NOTES: WHY GTD DOESN'T WORK

*Continued from page 31*

and point out that you deal with this in the Weekly Review — something else a lot of people never get around to doing. But that's another column.)

And then your boss walks in. Within a minute and a half you have now added seven edicts to your system. So you're not only capturing everything that your brain can churn up, but you also have to capture everything somebody else's brain can churn up and dump on you.

If you enjoy other people and like doing things to help, you probably are one of us who has trouble saying "no." So again, you start adding items to your system that other people come up with because you willingly accept the responsibility for it. It's called overcommitting. But at least it's captured. Of course, sometimes we overcommit because we just want to be nice and helpful, and sometimes we do it because we don't have a very good handle on how much we can really accomplish in a given period of time. It comes from having unrealistic expectations not only what we can do but how much time a particular task would realistically take a normal person. (Underestimators Anonymous. A Twelve-Step group with only seven steps.)

You might try to imagine how many things you would have down on your list if you took a full day, or even two, to go through every physical inbox, piece of paper, wallet, purse, smart phone, refrigerator cork board, bar napkin, plus a sweeping brain dump and put it all on a paper list or word-processing document. And I would be willing to bet that you would be shocked at how wrong you were. One of the times that I went through this exercise and got to "Inbox Zero" (GTD slang for having gone through all of your in boxes initially and either processing the items or capturing them in your system), I was stunned to find that I had over 650 entries. Sure, a large number of those items were on the Someday/Maybe list, but still!

And that, I think, is where a lot of people start — with an illusion of being "in control." And then after a few weeks, or maybe only a few days, of trying to work these lists, a feeling of being totally overwhelmed sets in and depression soon follows. And that depression leads to just giving up.

That's why I think GTD doesn't "work," at least the first several times some of us attempt it. It is all tied back to a misunderstanding of one of the fundamental concepts of the methodology — that of "ubiquitous capture."

Based on the nature of all of the unfiltered stuff that it seems to encourage us to collect on tour lists, I think a more accurate term might be "Ubiquitous Crap-ture." **IS**

Clif Notes

# Why GTD Doesn’t Work

BY CLIFTON OLIVER

And there you go. Without making it to even the first line of the article text, I have succeeded in alienating several hundred geeks. (“So what else is new,” I hear you say.)

For those of you who possibly do not know, GTD is a common abbreviation for Getting Things Done, developed by David Allen around 2005. It is currently one of the most popular time management methodologies in use, especially among geeks (possibly because of how glowingly geek websites like Lifehacker and 43 Folders describe it).

Over the last 40 years, I have tried numerous time management systems. As a matter of fact I have used GTD… numerous times. Believe it or not, I am back to using it now. So perhaps a more accurate title for this column might be , “Why GTD Only Works for a While.” Or possibly, “Why GTD Works But People Stop Using It.” I am guessing that this is about the tenth time I’ve come back to this system. Maybe when it comes to behavior modification I’m just a slow learner. Or I lack the meta-discipline necessary to discipline myself to use a particular discipline. But for whatever reason, I think I finally figured some things out about GTD.

One of GTD’s key concepts is that of “ubiquitous capture.” The idea is that anything you have rattling around in your head that you are trying to remember to do something about — be it an individual task or a series of tasks necessary to launch a new project — is effectively burning up processing cycles in your biocomputer. (Okay, they call it a “brain” like everybody else does. But you can see how the concepts translate to geek-speak.) In order to offload this stuff, you have to be able to dump it into a trusted system that can remember it for you, so that you don’t have to. This can be a Moleskine notebook, index cards, a pocket voice recorder, or any other device or collection of devices that will make sure it gets captured as it happens. I happen to use a MacBook computer, an iPad, and an iPhone — all running a product designed specifically with GTD in mind called OmniFocus that keeps everything synced between all the devices. Hey! I’m nothing if not thorough.

This is the first problem with this methodology. It encourages you to capture everything, all the time, everywhere. That sounds good in theory, but in today’s world of the Information Explosion, we just have way too much stuff flying at us. You’re sitting at your desk web surfing. You read something that talks about a new programming language, database, or maybe content management system. You think you might want to look into this in more depth. Capture it. The phone rings, and your spouse asks you to go by the dry cleaners on your way home from work and pick up the clothes. Capture it. You remember that you wanted to stop by that new gym that just opened and pick up some information about membership plans and check out their facilities. Capture it. (This is obviously your list we're talking about here, not mine.) The shop calls and tells you what this lastest repair bill on your car is. You decide it’s about time to buy some other vehicle. That’s more than one step. So that’s a project. The first action step in a project is to plan it out. Capture it.

You are getting the idea. Anything that is not just a passing thought but something that you must, should, or want to do that you cannot handle right now needs to be captured so you don’t lose it. This also includes things that you have not even decided to do but might want to do “someday/maybe.” I don’t know about you, but if I have to stand in line at the post office for five minutes, my brain is still ticking. By the time I get out of there, I can have at least ten more items stuffed into my GTD system.

There you see one of the first problems. There is an emphasis on capturing things for fear that we will forget them and miss a necessity, obligation, or opportunity. But there is little, if any, thought given to take a small amount of time to make a moment-by-moment decision of whether or not this item even needs to be captured. (Hard-core GTD-ers will probably want to jump in

*Continues on page 30*