$7.00 U.S.

INTERNATIONAL Spectrum®

THE MULTIVALUE TECHNOLOGY MAGAZINE | JANUARY/FEBRUARY 2014

Resolved

Do MultiValue Better

Also in the Issue:

- Google Glass – part 2
- Clear Messages Revisited
- 7 Requirements for ERP

intl-spectrum.com

The Premier MultiValue User Event

INTERNATIONAL Spectrum®

2014 MultiValue Conference

April 7 – 10, 2014

From application development and technical systems support to complex business and web development, we offer complete IT solutions.

The **International Spectrum MultiValue Conference** is the only place that provides diverse solutions and support for integrating and developing your Enterprise applications into the world of business and technology.

This annual event is the place to find the **knowledge**, **people**, **resources**, and **technologies** for enhancing your MultiValue (D3, UniVerse, UniData, Reality, jBase, Caché, QM, and OpenInsight) software applications.

Topics covered at the International Spectrum MultiValue Conference include:

- **Integration**
- **Database Management**
- **Mobile Integration**
- **Reporting and Data Presentations**
- **Project Management**

...and more.

The Wigwam | Litchfield Park, Arizona

www.intl-spectrum.com/conference

INTERNATIONAL Spectrum
THE MULTIVALUE TECHNOLOGY MAGAZINE

DEPARTMENTS

FEATURES | JANUARY/FEBRUARY 2014

From the Inside

Big Data is still a hot topic, even two years after the term was first introduced. Although, it seems people are still not sure what exactly they should be doing regarding Big Data.

I talked about the terminology of what "Big Data" was supposed to be to mean a few years ago. For those that didn't catch that From The Inside, here it is again.

Big Data references to the complexity, amount, and the management of a large quantity of data. If you look at the definition on Wikipedia, you will see it talks about "terabytes and petabytes" of data, but it also states "beyond the ability of commonly used software tools to manage and process within a tolerable time."

> *Big data is a term applied to data sets whose size is beyond the ability of commonly used software tools to capture, manage, and process the data within a tolerable elapsed time. Big data sizes are a constantly moving target currently ranging from a few dozen terabytes to many petabytes of data in a single data set.*

Reference: http://en.wikipedia.org/wiki/Big_data

Let's look at the concept of "tolerable time" and "software tools." At first glance you may say, well, I don't fall into that category. I can manage all my data in my programs within a "tolerable time."

Tolerable time is extremely relative due to the who, when, and why an enterprise has to deal with on a daily basis. Are you generating reports? Are you generating lookups and presentation for immediate action or batch processing? When does the user expect this information to be presented to them... Now or in 30 minutes?

The "Big" in Big Data is relative as well. I've seen articles on the Internet arguing about what constitutes Big Data sizes. Some define it in the amount of data being stored and the size of the database.

Let look at the classic example of sensor data. This data is usually highly structured and not very big per transaction (50 Bytes). But there can be large number transactions. If that is the case, 1,000,000 transactions would be about 48 MB. That is not terabytes and petabytes. It would take 21,990,232,555 transactions to reach 1 terabyte. (60,247,212 transactions a day for 1 year) While this may happen in your environment, in a normal business environment this is not likely.

I don't necessarily agree that Big Data represents either of these things.

Big Data is not about the size of your database but the complexity of the data you need to work with. The idea of Big Data was coined to address the problem of working with data that have complex relationships, and how to find trends and patterns to solve problems using large volumes and a variety of data.

We all have large amount of data in our systems that can provide useful information if we just know how to apply the Big Data concepts. The sheer volume and complexity of the data can make it hard for us to decide to approach the problem, let alone even know what questions to ask.

At the 2014 Spectrum Conference, on April 7–10 in Phoenix, we will talk more in-depth about Big Data and what questions to ask and how to get the information from the data you already have.

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

INTERNATIONAL Spectrum
JANUARY/FEBRUARY 2014

NATHAN RECTOR
President

CLIFTON OLIVER
Editor

TRACEY RECTOR
Layout

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.

GET CONNECTED.

Knowledge and Education for the MultiValue Professional.

About Our Professional Memberships

We are all busy in our day-to-day work and staying up-to-date with the current MultiValue technologies can be difficult.

Professional Memberships provide you access to knowledge, solutions, information, and code that you won't find in other locations.

Professional Membership Includes:

- Magazine in Electronic and Print
- Newsletter
- On-Demand Videos
- Live Webinars
- Discounted Conference Rates
- Research papers
- Case Studies
- Source Code

INTERNATIONAL Spectrum®

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

New Year Resolution

BY BRIAN LEACH

I am always uncomfortable when I hear unwarranted criticism of MultiValue systems by people who are ill-informed or whose exposure has only been to the sort of old, arcane MultiValue systems of which there are far too many still in existence. In my small way I do my best to promote MultiValue as a leading edge technology whose benefits, once understood, have a clear place in the pantheon of technologies available to today's organizations. There are core systems in many key players in major industries that depend entirely on MultiValue and that would simply not perform with other technologies. Since I also work with some of those other technologies, I hope that I can present an open-minded and coherent argument of the benefits and downsides, and can advise on where MultiValue should be deployed and where other models, sitting side by side the MultiValue offerings, can be leveraged to offer the best advantages. This led, for example, to an in-depth meeting just a few days back on the merits of MultiValue, NoSQL, and SQL for a large organization and how the different technologies can mesh to deliver the full scope of what they want to achieve.

Just as our systems can become ossified, so can our practices.

Horses for courses and no one size fits all.

At the same time, I am always more uncomfortable when I hear entirely warranted criticism of MultiValue practices by people who are well informed and taking a brutally honest look at our industry.

Take as an example a new development manager at one of the major UniVerse sites in the UK. He had come into an organization whose business runs on UniVerse and that depends on UniVerse for its daily operations and for its customer outreach through its web site. This is a large 24 hour operation where system performance is critical and promoting high levels of customer engagement is their main differentiator in their marketplace and a key part of their strategy in winning business.

And yet, he told me, he was amazed to find that the UniVerse developers did not know how to write unit tests, that all testing was performed manually, there was no automated build process, no automated code checking, deployments were sometimes packaged and sometimes manual, and no source code control. Far worse, there did not seem to be any understanding that this unacceptable in today's world. How could he, as a manager, have confidence in the systems under his control if he did not have sight of what was happening at every stage?

Walk into any responsible Java, .NET, or web shop today and what would you expect to find? At the very least you will see:

- Good source control to show a glance what has been committed, when and by whom.
- Automated build and CI processes to show when a commit has broken the build.
- Coverage statistics from the run machines to show how much of the code has been tested.
- Metrics on code complexity to show bad practices and potential code quality issues.
- A fully automated pipeline that links together the commit, build, test, quality assurance and deployment stages.

Continues on page 7

NEW YEAR RESOLUTION

Continued from page 6

These things are not luxuries. The core components — source code control systems such as Git and Mercurial, industry standard build managers like Jenkins and Bamboo, BDD, and acceptance platforms like Cucumber and FitNesse — are free and open source (commercial offerings like TeamBuild are also available). Organizations hiring professionals expect them to be experienced in test driven or behavior driven development and to be conversant with these tools. They are the daily bread for development.

How is it that so many MultiValue sites have fallen so far behind?

It's easy to see the progression from outside. MultiValue systems just work, so are frequently underfunded. People get used to the patterns of working that have seen them through in the past. It's all too easy to go from: "This is how we've always done it" to "This is how we do it" to "This is the right way to do it" based on history rather than active research into the best practices around. Just as our systems can become ossified, so can our practices.

There is good news. I've been working for much of this year at Travis Perkins, one of the largest UniVerse users. Travis Perkins is a Builders Merchant that owns a significant number of other brands in the UK, most of which have been assimilated over time into their UniVerse system. For those who like numbers: around 40,000 concurrent UniVerse processes spread over 3 machines cooperating over uv/Net; over 12,000 programs and 4 million lines of code making up a 20 year old system; and running a business with an annual turnover of over 5 billion pounds. That's a lot of concrete, and a lot of code to manage. It's a busy system — for most of the day each system is handling a million reads a second. That's a lot of work going on.

At this years' XP Days conference I had the pleasure of co-presenting an experience report with Nik Silver, an Agile Coach working at Travis Perkins, on the transformation we have seen there over the last year. This had been a traditional UniVerse site, with developers working away at a code face in an atmosphere of silence as part of small hard working and under-resourced functional teams. There was a code review process with applications support but no direct engagement with the business. There were experts in

Continues on page 8

NEW YEAR RESOLUTION

Continued from page 7

different parts of the system, so critical knowledge was siloed. There were no current coding standards, no source control, and only green screen editors. Despite this, they still managed to keep the business running, but through heroic efforts, and parts were in danger of becoming unmaintainable.

Today, that Travis Perkins is history. With new leadership they have restructured their IT department into Agile teams following the Scrum model. Teams now have control over their workload and collaborate directly with their product owners. Teams are cross functional, so there is true knowledge sharing. There is a buzz about the place that I have never seen there before. Out of this has come a pride and ownership — passing one of the Scrum boards someone has just pegged up a sheet, "We do not ship FINE." Just how far this has gone is proven by one of the teams wining the Scrum Team of the Year at the Agile Awards for 2013.

And it is not just the organization that has changed, but the UniVerse practices. Travis Perkins has embraced test driven development: even the "old hands" who — frankly — took some convincing have bought in and would now not go back to the old ways of doing things. We have introduced a CI pipeline that begins with individual developers creating automated tests and profiles for creating and tearing down assets before committing to git repositories and pushing their changes. This in turn kicks off an automatic git pull to a CI account in which the code base is rebuilt and all the tests are run. Failing and passing tests are reported via web and email, and only on passing runs is a separate and segregated release software account populated so no software can be released without integration testing. The next phase is to link this to a new packaging system currently being approved so developers can see where their releases have been installed and trace them through UAT, Pre-Production, and Live. We have started to investigate how to gauge cyclomatic complexity for UniVerse code so we can improve and measure code quality and have plans to automate acceptance testing.

All this has been delivered against a 20 year old UniVerse system and for a business that is still rapidly expanding. And TP is not alone in embracing this. Where I've been teaching Test Driven Development, I've found developers — even old hands like me — getting enthused and recognizing the advantages. They just hadn't realized that this can be done with MultiValue.

It can, and it should.

Perhaps your organization does an end of year retrospective, and perhaps not. But take a step back and review your year and realize that with a New Year comes a new challenge for all of us who work in MultiValue.

Do better. IS

BRIAN LEACH is an independent MultiValue consultant and tools developer based in the UK. Find out more at http://www.brianleach.co.uk or follow his blog at http://mvdeveloper.blogspot.com

Developing for Glass

Part 2

BY CHARLES BAROUCH

Our plan had three parts:

1. Hook up Glass to an Inventory Database,
2. Use the scanner capabilities to read product and box labels,
3. Create an X-Ray effect, where looking at the labels pulls up information about the inventory.

In Part 1, we covered my attempt to build a pure Glass application for inventory control. While Glass has all the parts I needed, it lacked the glue to make them work together. In Part 2 we will see how our plan can be done as an Android App and still work using Glass. The key is the true-but-false statement that Glass has a built-in full instance of Android.

It is true because it does have Android inside. It is false because there is no touch-screen equivalent. With Glass you have "tap" for select, "swipe forward" for scroll forward, "swipe back" for scroll back, and "swipe down" for cancel. Additionally, you have voice commands for such acts as "take a picture" or "google" a word or phrase. Much of the usability of Android is built upon the touch interface. Losing that bit of magic changes a lot. We can create our application in Android if and only if we embrace the limits.

Much of the usability of Android is built upon the touch interface. Losing that bit of magic changes a lot. We can create our application in Android if and only if we embrace the limits.

Inventory Database

I called Ladybridge and arranged for a copy of QM for my test. The problem was that my ISP doesn't allow command line access. Working with Martin, we developed a web install process that allowed me to install QM without needing the command line. Once I had a MultiValue environment, I loaded a small amount of data. Because QM, like most MultiValue databases, has the ability to be exposed as a web service, I knew that I'd have no trouble using it on the backend.

The idea was to set up a pair of subroutines. One, FINDBOX, to look up the inventory based on the barcode which was scanned. The other, MAIN, to act

Continues on page 10

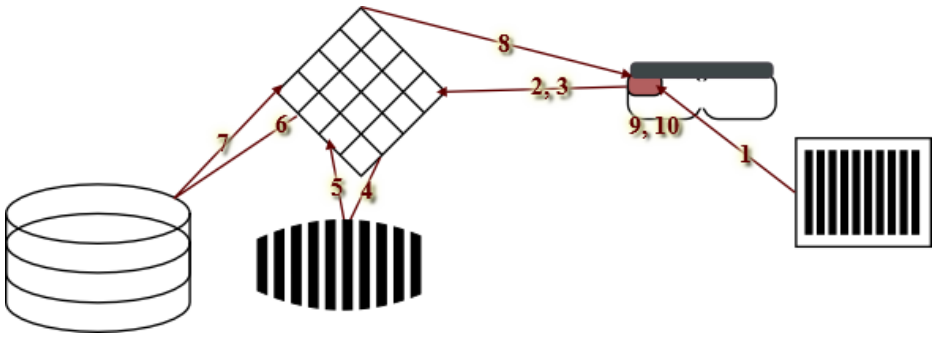


Fig. 1

GOOGLE GLASS PART 2

Continued from page 9

as a gatekeeper so that I could add other features in the future (fig 2).

Scanner

As I mentioned in the last installment, the Scanner in Glass wasn't going to do what I needed it to do. After consulting with a few Glass gurus, it was determined that the best course would be to send a picture and let the back end do the decode. As clever as our various MultiValue implementations are, I don't know of one which will tease a barcode out of a picture natively. We needed a piece to fit between QM and Glass.

Zxing (pronounced Zebra Crossing) <https://code.google.com/p/zxing/> suited the need. It is a backend application for turing a picture of a barcode into the code numbers. It supports all of these: UPC-A, UPC-E, EAN-8, EAN-13, Code 39, Code 93, Code 128, ITF, Codabar, RSS-14 (all variants), RSS Expanded (most variants), QR Code, Data Matrix, Aztec ('beta' quality), and PDF 417 ('alpha' quality).

With zxing in place, I would be able to meet my spec.

```
subroutine MAIN(RESULT,CMD,USER,OPTIONS)
call FINDBOX(RESULT,CMD,USER,OPTIONS)
return


subroutine FINDBOX(RESULT,CMD,USER,OPTIONS)
* CMD layout: find 9781906471859
INV.ID = oconv(CMD,'G1 1')
read  INV.REC from INV.FILE, INV.ID else
      INV.REC = INV.ID : ' not found.'
END
convert @AM to char(13) in INV.REC
RESULT = '<div class="data">' : INV.REC : '</div>'
return
```

Fig. 2 Our Code

X-Ray

The return trip was the next challenge. Just like with QM and zxing, the solution is the web. Here's my new roadmap (fig. 1):

1. Glass: User scrolls to my app and taps to activate it.
2. Glass App: App allows use to tap to take a picture and send it over the web.
3. Backend Web (via PHP): receives the picture.
4. Backend Web: calls zxing.
5. Zxing converts the picture and returns a number.
6. Backend Web (via PHP): take the returned number and calls the QM subroutine.
7. QM: looks up the number and returns the contents.
8. Backend Web (via PHP): takes the response from QM and sends it back to the Glass App.
9. Glass App: Like any other browser, it waits for the response from the server and then displays the results.
10. Glass App: Since we see the result inside the App, we can let the app take the next picture simply by accepting the next tap.

Wrap Up

I've yet to finish this masterpiece. Life continually intervenes. I now have every piece. All I lack is time. Since this approach will work on any Android device, maybe one of you will beat me to it and send me a finished copy of the app. There's no reason it couldn't be an iOS app, for that matter. We have reduced the front to: (1) send a picture and user ID, (2) display the response, (3) repeat or quit. This allows us to implement the app on nearly any hardware.

I'll continue to plot and plan with my Glass. Perhaps there's another article in the offing. IS

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

Pragma Systems and BlueFinity Expand SSH, mv.NET and UniVerse Access to Windows

Pragma Systems Inc., a leading provider of enterprise-class remote access and security software for Windows servers and mobile devices, collaborated with BlueFinity to allow BlueFinity's mv.NET application to support SSH access to Windows servers running applications on UniVerse. This represented the first combined utilization of Pragma's SSH Fortress, Windows and UniVerse all running seamlessly across the infrastructure.

BlueFinity's customer team previously deployed Pragma's Fortress SSH Server running under UniVerse and Windows. When a solution requiring the integration of mv.NET was needed, Pragma's engineering team (SSH expertise) partnered with BlueFinity (.NET expertise) to test various options and software updates until a satisfactory solution was found to incorporate the mv.NET application. The collaboration between Pragma and BlueFinity seamlessly delivered a fully secured functioning end to end system to the customer.

"The interaction with Pragma support was seamless, with both parties working together towards a mutually beneficial resolution to any problems encountered," said Mike Street, Support Analyst at BlueFinity.

"Working with BlueFinity to customize a solution for their customer expands the functionality of this system set up to other companies that could benefit from this partnership," said David Kulwin, CTO, Pragma Systems.

BlueFinity International has based mv.NET on the .NET technology from Microsoft, allowing organizations using MultiValue technology to utilize both the application development tools and the huge pool of .NET aware application developers in order to meet the significant challenges of an ever changing application development landscape.

FortressSSH Server is an enterprise grade secure shell (SSH), allowing secure remote access and high-performance managed file transfer using SFTP and SCP protocols. Multiple users can run any console/character application remotely which scales to accommodate more than 1000 concurrent sessions and is customizable to meet a customer's unique environmental and application needs. Pragma's Fortress SSH Server holds its own FIPS certification (#1500), is listed on the DISA Unified Capabilities Approved Product List (UC APL) and successfully passed rigorous DoD testing through JITEC.

About BlueFinity International

BlueFinity International, part of the Mpower1 group of companies, supplies leading-edge software development tools and consultancy services to the MultiValue database and Microsoft developer communities. Its flagship product - mv.NET – is a comprehensive solution for developers wishing to access MultiValue databases from within Microsoft's .NET environment. For more information visit www.bluefinity.com

About Pragma Systems, Inc.

Pragma Systems Inc. is a leading provider of enterprise class remote access and security software for Microsoft Windows & Windows Mobile platforms. Pragma is an industry leader in Secure Shell and Telnet technology and offers the most popular SSH and TelnetServer for Windows. The company's end-to-end solutions of servers and clients on the desktop and mobile devices provide highly secure access to corporations, supply chain, CRM, distribution and warehouse applications over wireless, Bluetooth, LAN, WAN and mobile networks. For more information visit www.pragmasys.com . ■

InterSystems Recognized in Gartner Magic Quadrant for Operational Database Management Systems

InterSystems, a global provider of advanced software technologies for breakthrough applications, recently announced it has been positioned as a Challenger in the recently published "Gartner Magic Quadrant for Operational Database Management Systems," by analysts Donald Feinberg, Merv Adrian, and Nick Heudecker.

"We believe our recognition in Gartner's first Magic Quadrant on operational database management systems reflects our clients' appreciation of our advanced technologies that enable breakthroughs in Big Data processing and analysis, as well as rapid application development," said Paul Grabscheid, InterSystems Vice President of Strategic Planning.

"InterSystems has always been at the forefront of cutting-edge database technologies," said Robert Nagle, InterSystems Vice President of Software Development. "Caché can ingest and process

Continues on page 12

Continued from page 11

an immense volume and variety of data at lightning speed. And embedded within Caché is a unique text analysis technology, called InterSystems iKnow, that analyzes unstructured data without requiring ontologies or dictionaries. We married that with our real-time analytics technology, called InterSystems DeepSee, making it possible to embed powerful analytic capability into transactional applications. With Caché, our clients can rapidly extract complex concepts and build cubes on all their data, both structured and unstructured."

Gartner defines the operational database management system (DBMS) market as "relational and nonrelational database management products that are suitable for a broad range of enterprise-level transactional applications, including purchased business applications such as enterprise resource planning, customer relationship management and customized transactional systems." In their report, Gartner stated "For this Magic Quadrant, we define operational DBMSs as systems that include support for new structures and data types, such as XML, text, audio, image and video content, in addition to traditional structures."

To view the full report, "Gartner Magic Quadrant for Operational Data Base Management Systems," by Donald Feinberg, Merv Adrian, and Nick Heudecker, October 21, 2013, please visit (intl-spectrum.com/s1066)

About InterSystems

InterSystems is a global software leader with headquarters in Cambridge, Massachusetts, and offices in 25 countries. InterSystems provides advanced technologies for breakthrough applications. InterSystems Caché is an extremely fast and massively scalable database system. InterSystems Ensemble is a platform for rapid integration and the development of connectable applications. InterSystems DeepSee and InterSystems iKnow are technologies for conducting real-time active analytics with structured and unstructured data. For more information, visit InterSystems.com.

Disclaimer:

Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability or fitness for a particular purpose ■

Westminster College Integrates LMS with Entrinsik Informer Reporting & BI

We recently partnered with Westminster College to present a live webinar that aired on Wednesday, November 20th, 2013. In case you missed it, you can download the webinar recording on our website.

Nichole Greenwood and Troy Derber from Westminster College present their own experiences with Informer BI & Reporting in the first of a series of customer-led webinars hosted by Entrinsik. We've provided a brief overview and some key highlights of the presentation below. We would like to thank Nichole and Troy for their excellent presentation!

Nichole Greenwood occupies an Institutional Research Role at Westminster College and co-presenter Troy Derber works in the IT department as a software developer and systems administrator.

Westminster College is a small liberal arts college with around 3,000 students, and has been using Informer since 2008 with almost 6,000 reports created. Implementing Informer has allowed the College to report off of their Ellucian Colleague and other databases simultaneously without the need for complicated data warehouses or cubes for instant access to the data they need.

Before Informer, reporting was done fully by IT using colon prompt to pull data from the College's databases; non-IT users did not have access to institutional data and relied on the IT department for the information they needed. Westminster chose Informer partly because it was the easiest and quickest to implement; users had instant access to data and reports with little time needed for training.

Most of Westminster's reports are written by a small group of power users, which can then be run and manipulated by end-users across campus. Many reports are automatically scheduled to be emailed out to end-users for quick, easy access to report results right in their inboxes.

Many end-users including faculty members lacked the time or detailed parameters needed to run reports. To solve this problem, Westminster integrated Informer with their LMS (Canvas) to help faculty access specific data right inside the LMS site. Within the Canvas interface, a "WC reports demo" tab was created, which links directly to Informer. In this tab, faculty can access specific, live

Continues on page 13

Continued from page 12

report data such as a photo roster of students in a course, average GPA of a course, student athletics, etc. To do this, Canvas grabs live data from a pre-specified Informer report, and custom-built charts are automatically updated. The College hopes to incorporate Informer dashboards in this process soon so that users can drill down into chart data.

Dashboards have been widely used in many other campus offices and several are distributed campus-wide. This has been especially helpful in visualizing student enrollment statistics (credit hours, undergrad enrollment by class, level, credit hours, etc.). Westminster hopes to further utilize Dashboards in future planning. Informer makes it easy to embed dashboards into another application (like Canvas) or directly onto a specific webpage by direct url link or html embed code.

The College estimates they have saved 5-6 hours of reporting time per day after implementing Informer. More users now have access to important data and have the ability to copy reports, change selection criteria based on department or school, and have almost instant customization. Informer makes reporting easier and faster; it's that simple!

About Westminster College

Founded in 1875, Westminster is a nationally recognized, private comprehensive liberal arts college with a broad array of undergraduate and graduate programs with more than 70 academic programs to choose from.Westminster's unique environment for learning prepares students for success through active and engaged learning, real world experiences, and a vibrant campus community. The College's location, adjacent to the Rocky Mountains and to the dynamic city of Salt Lake, further enriches the college experience.

About Entrinsik Informer

Entrinsik Informer is an agile reporting and business intelligence (BI) solution delivering ad-hoc reporting, self-service data analysis, and interactive dashboards in one complete solution. Informer's intuitive, web-based interface is easy to navigate, enabling users to access data from multiple sources to create and customize their own reports and dashboards on demand. For a demonstration or a free trial, visit http://www.entrisik.com/informer/request-demo, call 888-703-0016 or email sales@entrinsik.com. ■

BY SUSAN JOSLYN

Clear Message Initiative

"Danger! Danger!"

This warning was issued frequently from the tinny speaker box of the unnamed robot in the 1960's television series *Lost in Space*. The message communicates to a limited degree. It does successfully identify the general (dangerous) condition. It doesn't offer any information about the danger or advice about what to do, but the fact that it is a warning about danger is clear enough. It gets a bit better when specifically directed: "Danger, Will Robinson!" Still no indication of what is dangerous or what our intrepid Will Robinson should do about it. But it is more targeted and specific; it is Will Robinson who is in danger. Another oft-quoted warning message from these earliest — albeit science fiction — days of computing was, "Does not compute." This one gets top honors for useless information. It just doesn't get any more vague than that! As the years have passed, there has been much progress in technology, but not as much progress in message clarity. Bad error messages may, in fact, be the bane of the modern world.

Recently I was unexpectedly thwarted in the routine task of syncing my smartphone. An option-less dialog announced, "Function OpenFolder failed." Complete fail. I can venture a guess or two about the cause, using my own programmer expertise, honed deductive reasoning skills, and knowledge of the English language. Some folder has gone missing or corrupted. I wish I knew what might have changed. Or maybe if the message had indicated the name of the folder, or what the program was trying to do, more broadly when it encountered the problem, these clues would have given me something to try before that age-old catch-all solution — delete and reinstall the whole application. How much time do we waste on that particular activity?

A successful error message may or may not include humor, but it should definitely be informative.

Not only is my life filled with cryptic, inexplicable, counter-productive error messages, but I have come to realize that my own software is positively riddled with them! Until recently I guess I figured that getting the software to work was important and reporting when it didn't was not. Nobody reads these messages anyway, was my thinking. They'll call (or e-mail) me. And they do. But this is a self-perpetuating cycle! They might not have to turn to me if the error messages and warning they received made clear sense and offered an alternative.

To be fair to our ancestors, resources were different in the breaking dawn of the information age. Each letter-character required time, effort, and money to display (or print). Options for output were limited. Messages were necessarily cryptic. The limitations are mostly long gone, now, but the culture that had grown up around them has remained. Maybe, to our credit, programmers just like to be *efficient*. We can probably agree that a common generalization about programmer-types is that we (well, some of you) tend to use words sparingly. Or it may be that programmers have their own language and non-programmers just "don't compute." Certainly users of an application appreciate different things in an error message from what the application's designers may want to know. Many a true geek believes that the classic DOS error message BAD COMMAND OR FILE NAME is a thing of true beauty. It is elegant in its simplicity — and so flexible! It can be applied to such a wide range of circumstances — one size fits all! What it is not, however, is helpful.

Continues on page 15

CLEAR MESSAGE INITIATIVE

Continued from page 14

So, here we are. You, me, your mom, your kids. Surrounded by fast moving technology that is fraught with errors and very little guidance. If we are going to progress at this rate — as a society, as a nation ,or as an industry — we have got to get better at reporting and explaining failure. And we have to offer guidance that doesn't require translation!

What do we fix in figure 1? Are we, as programmers, *lazy?* Do error messages fall into the dreaded category of "documentation," to be avoided at all costs? Do we believe that we are somehow protecting our users from knowledge that they can't possibly comprehend? Or are we protecting something proprietary for ourselves? While it is true that keeping things mysterious will help us retain our all-knowing mystique, it will also keep us tied to the phone answering the questions that the error message could have answered.

When my own error messages are lame, the most likely reason is because I really do have this notion that no one will really *need* them. I have to put something there; it's just what you do (or there is a mandatory ELSE clause), but I don't really expect anyone to have to rely on the message. Like life insurance — you buy it, but you can't really think about anyone ever *really* needing it. Yes, I do know that is crazy. And it's not a conscious belief. But I think it governs my behavior to some extent, and I don't think I'm alone in this.

Blogger Harry McCracken theorizes in his *Technologizer* entry The Thirteen Greatest Error Messages of All Time, that error messages should be painful at worst and boring at best. He contends that, "They tend to be cryptic; they rarely offer an apology even when one is due; they like to provide useless information like hexadecimal numbers and to withhold facts that would be useful, like plain-English explanations of how to right what went wrong. In multiple ways, most of them represent technology at its most irritating. They're rarely helpful. Actually, they usually add insult to injury. But what would computing be without 'em?"

We can joke about it. We can wax nostalgic. An error message from a machine out of your past can transport you back in time. But what we need to do — as a community, as an industry, and as right-thinking humans — is make error messages and warnings better! If we want to provide truly great software — and we do! — but we can't make it flawless — and we can't! — we must make it fail gracefully. We must offer a path to usability even, especially, in the event of a failure. Thus, I contend, the secret to quality software and customer satisfaction is communicating clearly through error and warning messages.

Let's start a revolution! Are you with me?

Take a look at the classic example — the http (404) "File not found" error (fig. 2). It is inarguably the most common error message in the human experience. In its early incarnations it was fairly unhelpful (see figure). Grating. Confusing. Negative. Bam! Denied!

But it was customizable and some enterprising organizations took it further. In fact, if you would like to have a laugh, use Google Images for the phrase '404 File not found' and look at what people have done with the place! (Or go to errorware.com where you can buy clothing with this and other great error messages emblazoned on it!)

Continues on page 16

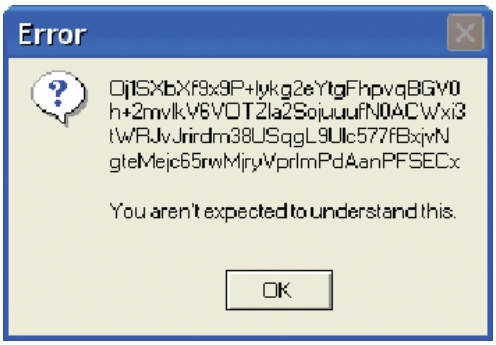


Fig. 1

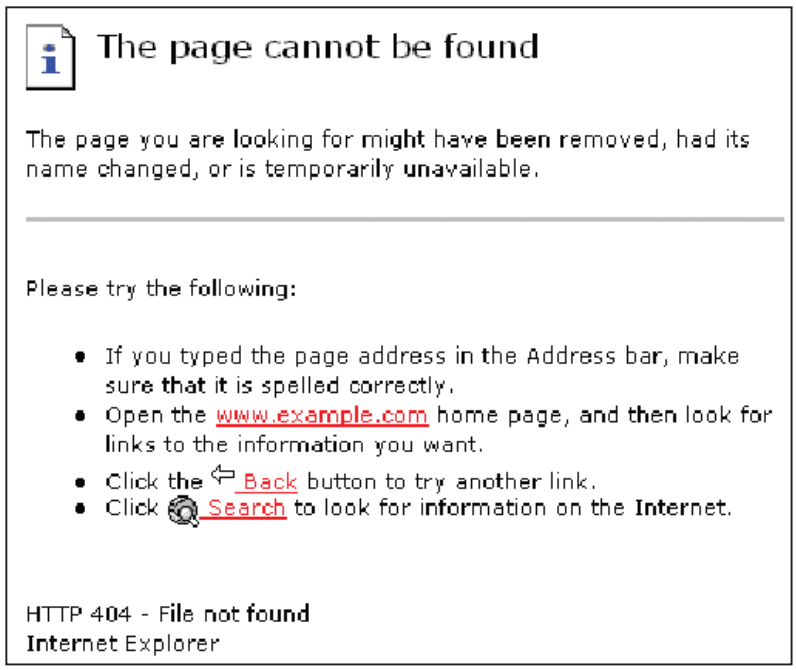


Fig. 2

CLEAR MESSAGE INITIATIVE

Continued from page 15

So, many web sites offered more polite versions. Some offered more information, included a graphic, a joke, or even a joke graphic. Those organizations have recognized a universal truth that is too often overlooked: when people hit an error and are prevented from doing what they meant to do, they will have an emotional response (and you know the emotion I mean). Throwing in something fun to make the user smile through their pain is not a bad idea, even if it doesn't help the user solve whatever problem they are facing.

A successful error message may or may not include humor, but it should definitely cover these points information:

- The severity: Whether this error critical, fatal, a warning, or just a reminder.
- The problem: what has occurred, what should have occurred, the reasons, and the impact.
- The application layer: what application (or layer) has produced the error.
- The action: What you can do about it right now, how you can find out more, and how you can prevent re-occurrence.

The challenge is to convey all of this information in a succinct 80 to 200 characters. It isn't impossible. Look what we're doing on Twitter with 140 characters.

```
87yr ago natl dads: "All equal."
War tests. Battle hallowed ground
> our words. We vow: dead not in
vain, govt of/by/for peeps here
4 keeps
```

Yes, that was Abraham Lincoln's Gettysburg addressed boiled down to a 140 character tweet. Twitter — and text messaging — have inspired a whole pseudo written language (as well as the dismaying notion of tiny URLS, but that's a topic for another day). These types of shortcuts would rarely be acceptable in a software application's error messages. So, we have to find a way to convey the information briefly but in more formal language, at least for now. This is probably best achieved by using a two part message. The main message text is like a heading. It must be concise, definitive, and structured. It should be instantly recognizable. Then a link, button, or second strike of the function key can take the user to a more elaborate explanation. Going back to that classic "404 File not found" error. Figure 3 shows how it has evolved in Microsoft's Internet Explorer 8.

Aside from the questionable graphic where the user is denied the entire earth, this is clear enough and easy to recognize by time you've seen it a few times. It is very general, but its saving grace is the magic "more information" arrow option. That second layer of help is where we can really communicate. Figure 4 is what you see when you click on that.

This addresses all of the elements, but it has so many maybes that it almost cancels itself out, and it is far too many options for a casual user to absorb. In the time it took to think up and type in all those maybes, at least in our easy-string-manipulation culture, we could

Continues on page 17

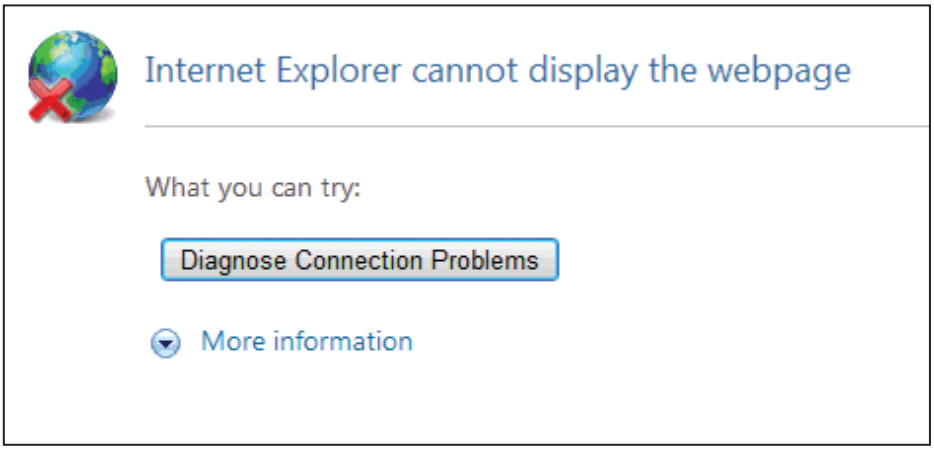


Fig. 3

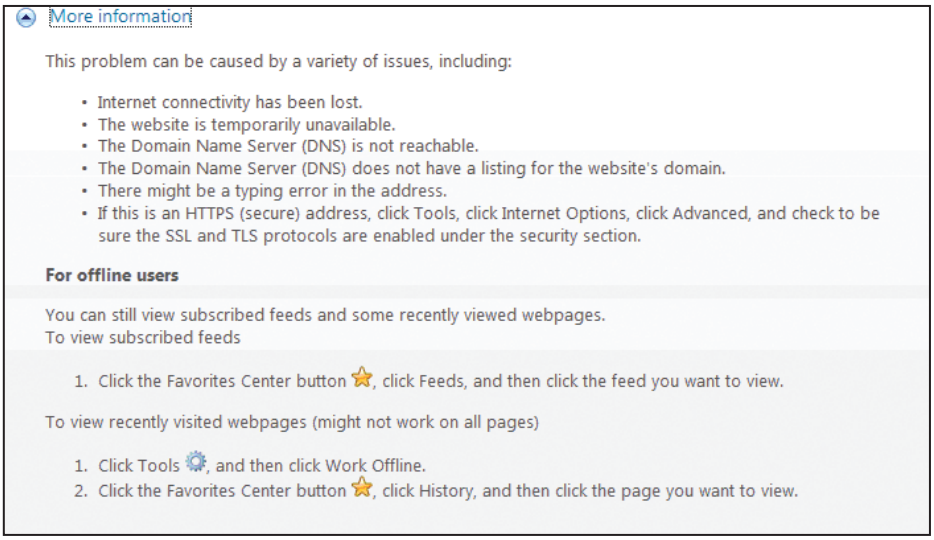


Fig. 4

CLEAR MESSAGE INITIATIVE

Continued from page 16

have figured it out a bit and at least limited the choices. This is where we (MultiValue/U2 programmers) can use the flexibility of our platform to separate ourselves from the herd. But I digress.

This message is better . But what makes an error message good? Yes, it should have all the pieces of information. But because that information is crammed into a small place and not really going to be read in depth, *how* it is organized and displayed is going to matter just as much as what it says. These are some starting points that can get the conversation started.

Standardize

There aren't any unified guidelines for error messages. Some application development environments have their own graphics and character-length limits. Microsoft has some general standards. Some standardization with other commonly used applications is helpful to the masses. But what is more important is that a standard be applied within the application.

Recognizable

If there is a main word or phrase that is commonly used it can be instantly recognized and absorbed. "Not on file" or "Required fields missing." Whatever the specifics — and there should be specifics! — that come after, these broader category-phrases will help. Using graphics to standardize the severity or layer of error is a very nice touch if that is available in the platform. Anything that makes it quicker to understand for our human users who understandably have very little patience for error messages.

Layer

We work in complex environments. When something fails, it could be something in our application, something in the application development environment, something in the underlying operating system, something with the network, or even something with the server. That could be the host, the client, or some server in the middle. It is a great help to identify the *layer* where a problem has occurred in the error message.

Syntax, grammar and punctuation and spelling

For the linguistic sticklers among us — and we know who we are — poor syntax, grammar, or punctuation can feel like the screech of chalk on blackboards. It is simple enough to check a style guide if you are not sure. A small typo can end up preserved like a fossil for all posterity.

Word order, tense, passive vs active

Passive? Nouns first? "Customer name must be alphanumeric." Active, lead with a verb? "Enter alphanumeric customer name." Or passive with the adjective descriptor first? "Alphanumeric is required on customer name field." "Decimals are not allowed in quantity field." Whatever you prefer, if you decide on it and stick with it, uniform error messages will be more quickly parsed and absorbed by the users.

If you use active/directive commands, be careful with the tone. According to some guidelines it is better to avoid accusative or admonishing tones. As rewarding as it might be to say, "You entered a bad name!" you can almost hear "jerk" at the end. Or, "Please don't try to use an alpha character in a date field." Again you can almost

Continues on page 18

CLEAR MESSAGE INITIATIVE

Continued from page 17

hear the exasperated again at the end. Expressing our own frustration as programmers only fosters bad feelings.

Specificity vs Generalization

Reusability is a great concept for development. But when it comes to errors, we must be careful not to generalize so much that our messages become useless. Many applications have been designed with error message databases that can be referenced by number. This can be a boon to standardization and a great way to provide lots more information for that second click. But we run the risk of starting to think of those error messages as precious resources again. Perhaps it feels sloppy to have a lot of similar messages. There are ways to combat the sloppy feeling and address the inefficiencies while still providing useful, specific information (variables in the error message, mainly). Or let go — let it be inefficient underneath so that it can be more useful on the surface! Messages need to offer specific information to the user. What may feel elegant and streamlined to the programmer is not always the most satisfying user experience. Take an extra moment when crafting an error message to see how much information you can give the user about the specific failure in terms that are meaningful to what the user will be trying to accomplish.

Let's start a revolution! Let's look like the best applications out there even when our interface isn't the flashiest graphic. Because even in a green-screen environment, we have the powerful and easy programming language to allow us to do that. If we remember what it is like to be that poor befuddled user trying to sync our smartphone, and if we assume that our error messages will be seen, that they will be needed, we can do better. Above all, let's remember that our error messages are a reflection of ourselves and of the quality of our workmanship!

Linked IN: Want to contribute ideas and suggestions – perhaps gain insight about what others are doing? Join us in the newly created Linked In group "Clear Message Initiative". 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.

BY NATHAN RECTOR

7 Must-Haves for every (ERP) Enterprise Application

ERP applications are made to serve a specific purpose — running your business. People use them every day, and their value lies in the depth of service they provide your business.

While they help run your business, most ERP packages these days are in really bad shape. They can be the least friendly and most difficult to use.

But still, people have to use them every day. And if they don't like them, they gripe. The huge battle between functionality vs. cost to replace them suddenly looks pretty small compared to dealing with unhappy employees. No enterprise application can last forever in the face of employee dissatisfaction, regardless of its value in the enterprise.

All those employees, including C-level management, have consumer items that they are comparing the business ERP application to: Cell phone, big screen TV, PDA, tablets, etc. They begin to expect the company's ERP to employ the same features and functions, and wonder why they don't.

The whole concept of any software application, ERP or consumer, is to increase a user's productivity, or make their life easier. If it makes someone's job harder, or relies on the user to do things in a specific order, then the software application has failed the business.

Your ERP application should be the hub of all your data and business, not just another system where data exists.

Here are seven things that every ERP application should have going forward. If you do not include the majority of these, if not all of them, then your application is going to have an uphill battle in the next five years.

1 - Choice of UI Delivery Model

It is no longer about the User Interface (UI) , and what it looks like. It is all about the User Experience (UX). Part of the UX is providing the user a choice of interfaces: Desktop GUI, Web/Browser, or Mobile/Tablet. But that is not all of it.

A user expects the same experiences in the ERP application as they have in their consumer products like QuickBooks, Microsoft Office, and their local bank. They want to be able to do their job, or access their information, from where ever they are and from whatever they are in.

They demand the ability to do things their way, and they expect the ERP application to do just that.

If part of their job is to be notified that something needs to be done, they want to choose their preferred notification method: Email, SMS, Push Notification, Social Media, or Instant Message. Or any combination of those.

If part of their job is to manage things away from a desk, then they want to access the information they need away from the desk by using mobile devices.

They want to use their own cell phones, their computers, and work from home just like they work from their desk at the office. An ERP application should provide the user a choice so they can do their work when and where they want, in the manner this is most productive for them.

2 - Collaboration

Everything is about Social this and Social that, but when it comes down to what people really want to do, it is collaboration. Let's talk, let's share. This isn't just internally. It is also with vendors and customers.

Vendors would like to see your sales information for their products. They would like to share with you the orders and status of their products. Their whole goal is to encourage you to buy more, if they can provide you with information about why a product will sell better in your area than another. But they need information to make these recommendations.

Continues on page 20

7 MUST-HAVES FOR EVERY (ERP) ENTERPRISE APPLICATION

Continued from page 19

Accessibility of data by your suppliers and customers through self-service portals and feeds allows them to see more value in your company than in others. Not only that, but by providing sales information to them, many times they can provide you better costs because they know what you are likely to sell or manufacture in the future.

3 - Flexible and Modifiable Platform

This has always been a key feature of our MultiValue ERP applications, but in many ERP applications, modifying and enhancing can't be done, at least not done easily.

Even the best, more flexible, ERP package won't fit every business out of the box. If it did, then the setup and configuration would be so complex it would become almost unworkable. Most ERP packages only handle 80% of what your business needs "out of the box." The remaining 20% is the difference between you and your competitor.

Companies should not sacrifice that 20% (it's your profit margin) by having to conform to the way an ERP packages wants you do to do business. The ERP package should be conforming to the way you do business.

Flexibility doesn't stop at the ERP software. The business should be in charge of deciding what hardware they want to run their ERP package on, not the ERP software package. If they want to run on in-house hardware, then let them. If they want to outsource to a Cloud server, then let them. If they want to use Windows over Linux, or AIX over Linux, they should have that option.

4 - Push Based information

Traditional ERP applications require a user to ask for the information they need to do their job. Whether this is working with a menu tree, or favorites and short cuts, it all comes down to the user having to ask for information.

ERP should be pushing the information to the user when something needs to be done, not requiring the user to constantly pool or ask for it from the computer. This concept has been what is successful about social media — push the information to the user, do not wait for the user to enquire what the status of something is.

5 - Anytime, Anywhere access

Mobile, Mobile, Mobile… If the user does not have access to the ERP information away from the their desktops, then your users are not being productive.

Users now expect to be able to see information on their tablets, smartphones, and laptops. This goes back to UX (User Experience) as well. While users now expect this, most ERP applications don't even have a basic read-only mobile interface.

Most don't even have a web portal to ERP information or generated reports.

6 - Improved Business Intelligence

Business intelligence is a must, and we have been doing this for years, but they have been simple, static, and "here is the answer, don't ask for more" type reports. The new C-level execs are now very computer literate. This means they expect to be able to take the raw information from these reports and run it through the BI tool of their choice.

At the bare minimum, they want to be able to manipulate the reports and information provided by the ERP system to a limited degree: add a new calculation, filter the data, sort the data, export into Excel.

7 - Works with Existing Applications

It's a fact of life, there is no such thing as only one application that does everything these days. Most enterprises have a minimum of three applications that help run the business. Most are specialized routines that do specific tasks, but they are "islands" unto themselves most times.

It is the ERP application's job to exchange data with them so that the user doesn't have to do double enter or live with "out of date" information. Your ERP application should be the hub of all your data and business, not just another system that data exists in.

It's important that it can exchange information, even if it's one-way.

Most of these points are easy to implement, and do not always require you to purchase not software. Spending a few hours to implement even one of these to a very limited degree would provide a big bang and boost to your IT departments.

I will repeat this, because it the best way to think of your ERP application:

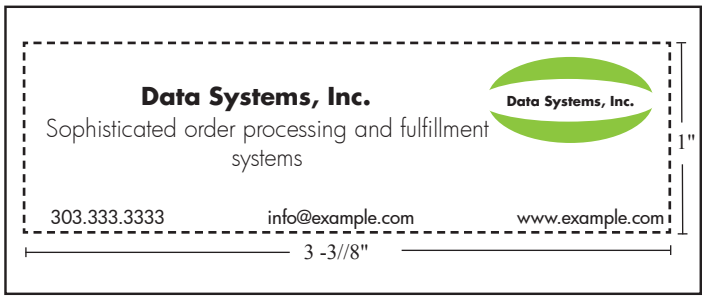
Your ERP application should be the hub of all your data and business, not just another system that data exists in. **IS**

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

MARKETPLACE

ADVERTISER INDEX			
AccuSoft Enterprises	18	Ladybridge Systems, Ltd.	7
Brian Leach Consulting, Ltd.	17	PICK Programmer's Shop	8
Execu-Sys, Ltd.	17	SJ+ Systems Associates	10
HDWP	24		

HIGH IMPACT 4-COLOR DISPLAY AD



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

MARKETPLACE

ACCOUNTING

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

COMPLIANCE

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

CONSULTING

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

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

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

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

Kore Technologies
www.koretech.com | sales@koretech.com

Norcron Inc. (U2/SB+ VAR)
www.norcron.com | sales@norcron.com | 404.459.6500
U2 & SB+ programming experts, Masterpack sales and support

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

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

CREDIT CARD PROCESSING

MV Tech Services Inc.
http://www.mv-tech.net | tom@mv-tech.net | 952.474.3795
Process credit cards directly from any MultiValue application.

DATABASE

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

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

REPORTING

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

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

TERMINAL EMULATOR

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

WANT TO SEE A SPECIFIC TOPIC?

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

E-mail: nathan@intl-spectrum.com

WANT TO WRITE?

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

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

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

E-mail: editor@intl-spectrum.com

LETTERS TO THE EDITOR

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

E-mail: editor@intl-spectrum.com

BY CLIFTON OLIVER

Do MultiValue Better? How?

Well surprise, surprise. I bet long-time readers never expected to see this column, let alone the current lead article of the magazine, be about New Year's resolutions. If you look back in the last two to three years of January-February issues at Clif notes, you will see that I have a habit of poking fun at the idea.

But for this issue, frequent contributor Brian Leach sent in a piece that struck a chord. Nathan and I got to talking and speculating on what it would be like if everybody in the MultiValue community were to make one, or possibly more, resolutions about something they would do in 2014 to help make sure that MultiValue continues to not only survive but thrive in the second half of this decade.

As we were kicking this idea around, we remembered an article that Susan Jocelyn had sent us and we published in theJanuary-February 2011 issue. That article was about what she referred to as her Clear Message Initiative. It seemed to us to the entire idea was something that could help start the creative thinking juices flowing of people looking for things to make resolutions about. So we asked her for permission to reprint it in this issue.

And to further tie into the topic, I'm going to step away from my usual tongue-in-cheek poking fun at people who make resolutions and break them within a week and make a few comments and perhaps several resolutions of my own.

When I was thinking about what kind of resolutions for MultiValue I would make, I realized that a number of them are simply not practical (yet) because of my career. I suspect that many of you will find yourself in the same boat of saying, "I resolve to… No, I better not."

For example, I would like to make the resolution, "I will never work on a green screen program again." Well… I really can't say that. I'm an independent consultant. Those of you have never been self-employed might think that means that I don't ever have to work on anything unless it is unique, interesting, and most of all fun. If most of the independent business people I know followed through with that attitude, they would quickly find the bank repossessing their house, losing their car or the ability to put fuel in it to get from point A to point B, and looking for two large cardboard boxes and an unoccupied space under a bridge so they could set up their new two bedroom condo. Sorry. I am not about to turn away paying work just because I don't happen to care for it or agree with the manner or style in which it is done. I am much too fond of the habit of having regular meals (as you can tell).

Or what about, "I will never again code in uppercase-only." Right. How do you think that's going to work for me? After all, there are million of lines of uppercase-only code running in organizations who might be able to use my help. See the arguments given in the paragraph above. Besides, I derive a great deal of satisfaction from being able to help organizations fix their problems, streamline and modernize their processes, and achieve new goals. That frequently involves working on legacy code written in all uppercase running green screen interfaces. That's just the nature of the beast.

So I don't want to make resolutions that I'm going to have to turn around and break in order to continue making a living at my chosen career. In many of your cases, you're not going to want to make resolutions that if you follow through on them are going to almost guarantee that you will get fired. (Of course, in thinking about your resolutions, you may decide to look for position with some other company who has more enlightened upper management. For an example of this, read this issue's lead article. It will give you hope.)

I frequently follow the discussions that people have on groups on various plat-

Continues on page 24

CLIF NOTES: DO MULTIVALUE BETTER? HOW?

Continued from page 23

forms – Google Groups, LinkedIn, etc. Many times these threads get very involved and passionate. I don't often participate in them because most of them fall into a reiteration of things I've been hearing since the 1980s or before. And after more than 30 years, I have nothing new to add to the conversation – until recently. In reviewing a number of these threads, there is one thing that becomes very clear to me.

MultiValue is not obsolete. But many MultiValue developers are.

I can't always change upper management's mind about modernizing their legacy applications. Some people don't want to change their minds. Their heads are made out of stone. Or they point to the uppercase non-object non-structured spaghetti code and the telnet sessions emulating green-screens as proof that MultiValue is obsolete and will be "out of this company within two years." (And then they spend ten years and hundreds of thousands of dollars to achieve that goal — many times with inferior results.) So be it.

I can't always change MultiValue developer's minds about upgrading their skills, preparing demonstrable prototypes, and doing whatever they can to actually use modern techniques and methodologies. Many times I can't even convince them to learn where the shift key is on the keyboard. But I can make sure that both management and developers have access to educational material, tools, and examples that demonstrate the use of modern MultiValue techniques and methodologies.

And I can try to lead by example, even in those projects where the MultiValue shop is stuck in the last century.

So here is my set of resolutions for 2014 to try to "Do MultiValue Better:"

- I will remain positive about the future of MultiValue. When people say MultiValue cannot survive, I will ignore them. When people say they want to see how MultiValue can thrive, I will show them.
- I will continue to update my own skills so that I can practice what I preach.
- I will use test-driven development techniques whenever and wherever I can.
- I will conscientiously conform to the objectives of the Clear Message Initiative.
- I will seek out developers who are using MultiValue in a modern way and encourage them to share their stories and techniques — both the successes and the failures — so others can learn from them.
- I will continue to try to help those organizations and individuals who want to bring their legacy applications into the 21st century to do so.
- I will endeavor to lead by example so others may also learn.

I'll keep notes so that next New Year we can get together and see where I was successful and where I screwed up.

So, what are you going to do in 2014 to "Do MultiValue Better?"

If you want to share your ideas with us, send them to me at editor@intspectrum.com or wco@oliver.com. We look forward to hearing from you! **IS**