



BIANNUAL
JOURNAL OF
VERYANT AND
isCOBOL

WHAT ELSE IS
NEW IN 2023R1?



- Added compatibility with Micro Focus, Acucobol, IBM, DB2, ProCobol, and VAX/OpenVMS
- Linkage sections with variable number of arguments
- Converting your data charset and supporting national characters to increase international compatibility
- MASS-UPDATE for Windows for smooth redraws
- Bitmaps on table-view style tree-view controls
- HTTPClient logging



veryant

NEWS

THIS ISSUE

1. 2023R1 Release News 2. Modifying your UI for mobile phones and tablets
4. Data Replication with c-treeRTG 8. Docker Containers – part four 10
Documentation Highlights 11. It's just COBOL, not Mainframe Programming
12. Tips & Tricks//New Videos and KB's 13. Last page

2023R1 New Developer Options

The new 2023 R1 release gives developers more options with a VSCode extension, a native Vision file handler, the ability to load test WebClient, and a better way to manage c-tree's data replication.

Visual Studio Code is a popular development environment for COBOL, and Veryant now has an isCOBOL extension that programmers can use to make it easy to edit their source-code in a familiar environment.

ACUCOBOL's vision file format is a popular one, and can be hard to move off of for various reasons. Now isCOBOL can handle Vision files as easily as Jisam and c-tree, with its newest integrated file handler, VisionJ.

The WebClient continues to mature into a robust tool to run your COBOL applications in a web browser or encapsulated in a web page with minimal or no changes to the programs. Now we've added a test tool that can automate your test cases, and run multiple runs for load testing..

c-treeRTG's data replication is easy to use and provides needed data stability and the ability to continue running your application when your server goes down. We've added a graphical, browser-based replication manager called Memphis (see "Data Replication with c-treeRTG")

Modifying Your UI form

Mobile Phones and tablets

Now that you can use [WebClient](#) to run your applications in browsers, there are some changes you can make to improve the user's experience.

When you look at your application running on a smaller screen like a tablet or smart phone, it might look crowded and tiny, making it harder to use. But we've made it easier for you to make some modifications so your application can look good on different screen sizes. These include switching to a hamburger menu, removing the title bar that makes it look like an application and not a web page, applying a layout manager with one configuration variable, and getting information about your environment.

Check your environment

If your application is running in WebClient, it may be trying to display on a smaller screen. Use the [C\\$GETRUNENV](#) routine to know where the application is being executed (thin-client, WebClient, standalone).

```
call "C$GETRUNENV" GIVING env-code  
if env-code = runenv-web-client
```

Undecorate your screen

If you're running in the WebClient, we advise you to set the [undecorated](#) and [maximized](#) styles. This will remove the frame and title-bar from your application, and expand it to fill the browser window no matter what size the browser window is. Your application will look more like a true web application than a desktop application.

Get monitor information

[C\\$MONITOR](#) returns the size of the monitors in pixels, which will tell you the size of the environment your program is about to be displayed on. For Thin client and Fat clients, it will also tell you how many monitors are available and which one is the main monitor. This can help you save resources downloading just the right font, and give you control over where your application displays.

Using the hamburger menu instead of the classic menu

It's more common now to see hamburger menus rather than menu bars on web applications. A hamburger menu is three lines that can hide the menu and expand when clicked on. In isCOBOL, it's easy to change your menus.

The code and logic to declare a hamburger menu is the same as a menu bar; the only difference is the op-code used to create it. The menu bar uses WMENU-NEW for the op-code, the hamburger menu uses the [WMENU-NEW-HAMBURGER](#) op-code.

Once you change the op-code to the hamburger menu, you have a lot of options to customize the layout using the op-code [WMENU-SET-ATTRIBUTE](#), including colors, fonts, size, position, and icons.

Apply a layout manager

A layout manager will resize your screen and controls with the size of the window. We suggest one of two options: [LM-ZOOM](#) or [LM-RESPONSIVE](#).

LM-ZOOM can be set in your configuration file to be applied to every window that doesn't already have a layout manager assigned. It will resize and move controls and fonts to fit in the window.

LM-RESPONSIVE takes a little more effort, because it needs to be applied to each control. But there are some screens that would look much better with this type of layout manager. To see this in action, look at the "RESPONSIVE" sample in issamples (under the 2019R1 new features section).

Uncrowd your screens

Sometime a screen is just too complex to display well on a small screen. This is the time to modify these screens, and use LM-RESPONSIVE to help redesign them for different screen sizes.

Do it the easy way

At Veryant, we want to minimize the code changes you have to make to update your application. So here's how to quickly make some of the changes outlined above:

Use code injection to have the compiler make changes for you. For instance, setting the variable below in your compiler's configuration file will make every window undecorated and resizable and use the LM-ZOOM layout manager

```
6 iscobol.compiler.gui.window.defaults= \  
7   undecorated, \  
8   resizable, \  
9   layout-manager lm-zoom  
10
```

Switch every menu to a hamburger menu with this line added to the beginning of your application

```
call "$menu" using wmenu-set-attribute  
                    "menu-bar-flavor"  
                    "hamburger"
```

1. Maximize your window with this line after displaying the window (changing it to the name of your window handle)

```
modify win-handle action action-maximize
```

Data Replication with c-treeRTG



By Fabrizio Cesura
Senior Support Engineer

Keeping data synchronized across multiple servers allows you to have consistent backup copies of the data and be able to switch to an alternative server in case the current server becomes unavailable.

The c-treeRTG server includes a plugin for data replication management. You can enable this plugin in the `ctsrvr.cfg` configuration file by removing the semicolon that comments the following line:

```
;PLUGIN ctagent;<libraryname>
```

The configuration of this plugin is managed with the following files installed with the product in the config folder:

File	Scope	Sample content
<code>ctagent.json</code>	Main configuration file. It includes the list of replication agent configuration files	<pre>{ "detached": true, "configurationFileList": ["../config/replication/ctreplagent1. cfg"] }</pre>
<code>replication/ctreplagent1.cfg</code>	Replication agent configuration file	<pre>unique_id REPLAGENT1 target_authfile ../config/replication/target_auth.set target_server FAIRCOMS@localhost source_authfile ../config/replication/source_auth.set source_server FAIRCOMS@SourceIP batch_size 8192 read_timeout_ms 5000 log_file_name ../config/replication/replagent1.log exception_mode transaction file_filter <../config/replication/filter1.xml replicate_data_definitions yes</pre>

File	Scope	Sample content
replication/filter1.xml	List of files to be replicated. Wildcards are allowed	<pre><replfilefilter version="1" persistent="y"> <file status="include">*.dat</file> <purpose>create_file</purpose> <purpose>read_log</purpose> </replfilefilter></pre>
replication/source_auth.cf	Credentials for the connection to the source c-tree server	<pre>USERID ADMIN PASSWD FAIRCOM</pre>
replication/target_auth.cf	Credentials for the connection to the target c-tree server	<pre>USERID ADMIN PASSWD FAIRCOM</pre>

The replication process is based on transaction log files: the replication plugin reads the transaction log files to understand which data modification has been performed on one server and requests the same data modification to the other server. While programs perform i-o on the c-tree server, the same i-o is replicated on the other c-tree server.

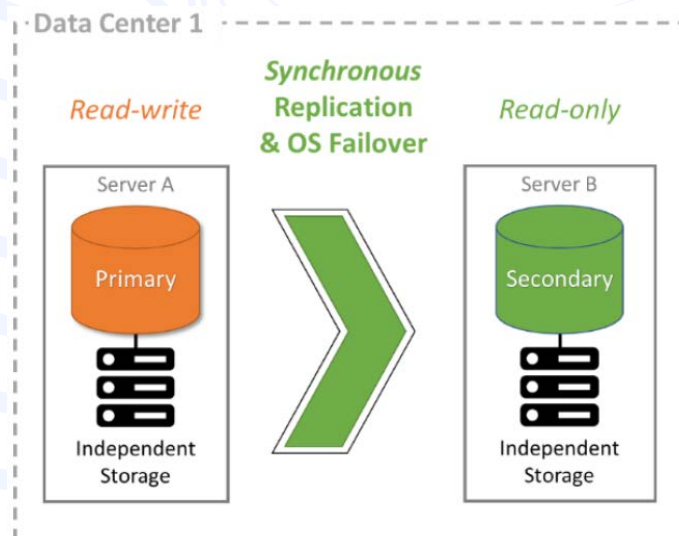
Only files with transaction logging enabled are replicated.

There are two types of data replication:

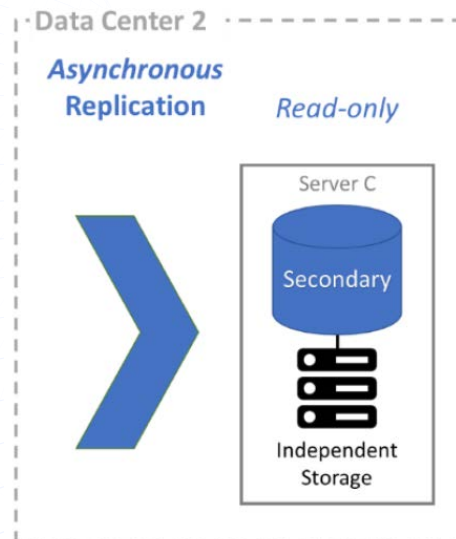
Asynchronous replication – replicates data between two servers through a non-blocking process. This allows the primary server to process data as fast as possible without being slowed down by a secondary server or the network.

Synchronous replication – uses a blocking process to replicate data between servers to ensure both servers have persisted a transaction to their transaction logs before the primary server reports back to the application that the transaction is committed. Synchronous replication is required for high availability because only it can guarantee data is always the same on both servers at all points in time. Because it is blocking, it is not as fast as asynchronous replication.

High Availability



Disaster Recovery



The replication type is set in the `ctreplagent1.cfg` configuration file. By default an asynchronous replication is performed. To perform a synchronous replication, remove the semicolon that comments the following line out:

```
;syncagent yes
```

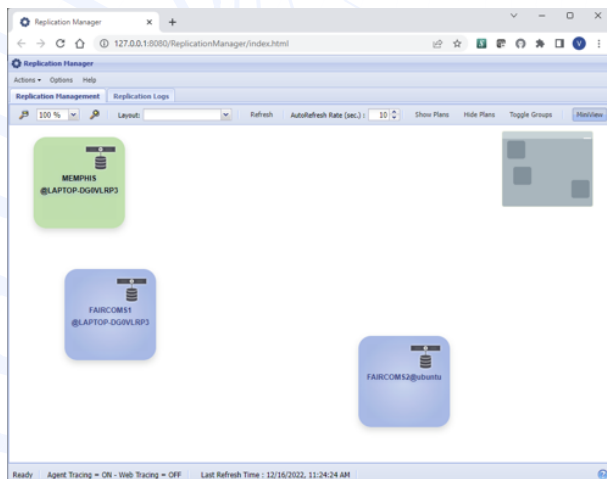
Once configured, the data replication starts along with the c-tree server.

The data replication activity is traced in a log file named **ctreplagent.log**.

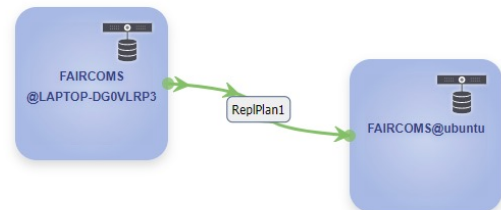
Use Memphis for Management

Since version 3.0.2.783 distributed along with isCOBOL 2023 the best way to configure and monitor the data replication is with the new graphical user interface, called “Memphis”, implemented in HTML5.

After starting the executable file named “Memphis” and the c-treeRTG servers in the network, you can open a browser and enter the URL: <http://server-ip:8080/Replication-Manager> (or <https://server-ip:8443/ReplicationManager> to use the HTTPS protocol). You will see the Replication Manager’s application with list of available c-tree servers.



In the Replication Manager’s application, you can drag a “Source” server over a “Target” server to create a Replication plan, based on a Publication where you can define if the replication is for SQL tables, ISAM files or both. You can also filter ISAM files included in specific folders that need to be included or excluded from the replication. The last steps of the configuration process are: Subscribe to the Publication, choosing if the replication is unidirectional or bidirectional, and Deploy the replication plan. After the Deploy, if the replication has started correctly, the line connecting the two c-tree servers turns green.



An overview of the current replication status and total amount of commits, records added, updated or deleted, can be activated from the Activity menu item of the pop-up menu of the deployed plan.

Plan activity for plan : ReplPlan1

Options

Time Frame: 1 Hour

Start Date: 12/16/2022

End Date: 12/16/2022

Start Time: 11:17 AM

End Time: 12:17 PM

Refresh

Auto Display Refresh Rate: 0 sec.

Dump

Details

Statistics Exceptions Function Timing

Page 1 of 1

Snapshot Status : Running

Stop

Collection Rate: 5

Displaying 1 - 28 of 28

Identification		Replication				Thread Counts			Transaction Throughput			
Type	Time	Paused	Aborted	Failures	Latency	Analyze	Apply	Commits	Add	Update	Delete	
20 sourceToTarget	12/16/2022 12:15:20 PM				0	1	8	3	2	0	1	
21 sourceToTarget	12/16/2022 12:15:30 PM				0	1	8	3	2	0	1	
22 sourceToTarget	12/16/2022 12:15:40 PM				0	1	8	3	2	0	1	
23 sourceToTarget	12/16/2022 12:15:51 PM				0	1	8	3	2	0	1	
24 sourceToTarget	12/16/2022 12:16:01 PM				0	1	8	3	2	0	1	
25 sourceToTarget	12/16/2022 12:16:11 PM				0	1	8	3	2	0	1	
26 sourceToTarget	12/16/2022 12:16:21 PM				0	1	8	3	2	0	1	
27 sourceToTarget	12/16/2022 12:16:31 PM				0	1	8	3	2	0	1	
28 sourceToTarget	12/16/2022 12:16:41 PM				0	1	8	3	2	0	1	

Close



The final part of a four-part series on isCOBOL and Docker containers, this article is a step-by-step guide to accessing shared memory while running more than one container. From Senior Support Engineer, Valerio Biolchi

A shared memory segment is a chunk of memory that is shared between multiple processes.

Docker containers are allocated 64MB of shared memory by default.

Communication between the isCOBOL Server and the c-treeRTG Server with shared memory running on different containers

Prerequisites

The prerequisites on a Linux 64bit computer (where the docker is built) are :

- The Docker engine (<https://docs.docker.com/engine/install/>) installed from a repository
- A licensed isCOBOL SDK environment
- An isCOBOL Server running in Docker having ISAPPLICATION sample setup and running. See instructions in the first article of the series, [here](#).
- A c-treeRTG Server Docker on Linux 64bit box (see instructions in the third article of the series, [here](#))

Run the c-treeRTG Server container

```
docker run --name ctreeserver -d -v shm:/tmp/ctreedbs -v /isapplication/data:/isapplication/data  
-p 5597:5597 --ipc=shareable ctree
```

`-v shm:/tmp/ctreedbs`

The shared memory protocols tokens are shared

`--ipc=shareable`

Enables memory sharing across the container

`/isapplication/data:/isapplication/data`

Mounts the host directory, /ispplication/data, to the /isapplication/data in the container

The c-treeRTG Server started by this container allows TCP/IP connections thru the exposed default port 5597, in addition to shared memory.

Communication between the isCOBOL Server and the c-treeRTG Server with shared memory running on different containers

Run the isCOBOL Server container

```
docker run --name iscobolserver -dit -v shm:/tmp/ctreedbs -v /isapplication/data:/isapplication/data --ipc=container:ctreeserver -p 10999:10999 iscobol
```

`-v shm:/tmp/ctreedbs`

The same shared memory protocols tokens shared by the “ctreeserver” container above

`--ipc=container:ctreeserver`

Enables the ability to join another “shareable” container’s IPC namespaces

The isCOBOL Application Server started by this container allows TCP/IP connections thru the exposed default port 10999, in addition to shared memory.

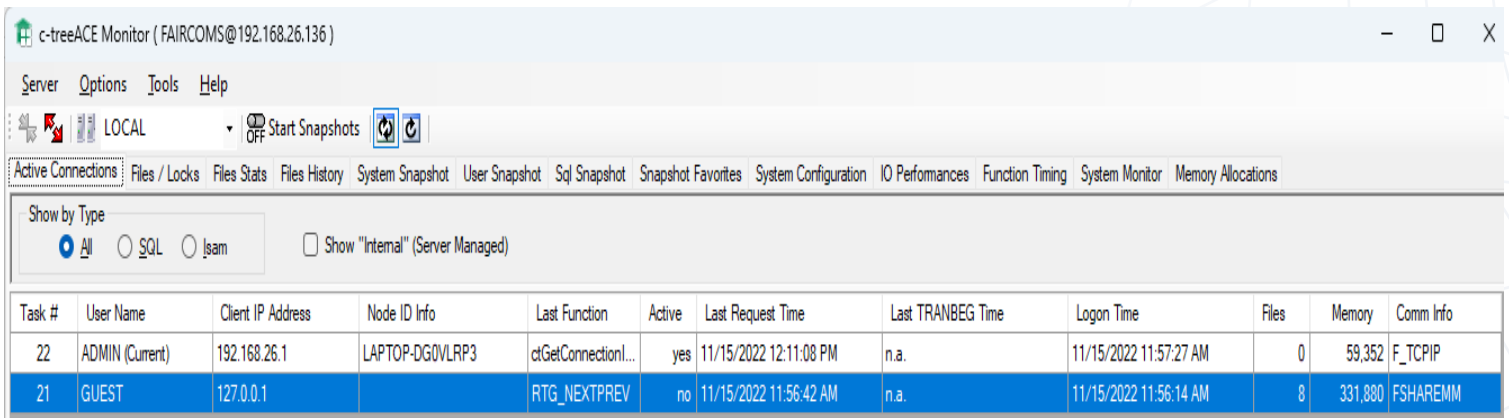
Execute the IO_INDEXED sample program a client PC

```
iscclient -c runtime.properties -hostname 192.168.26.136 IO_INDEXED
```

This command will run the IO_INDEXED program in the “iscobolserver” container and work with the c-tree indexed data files from the “ctreeserver” container thru shared memory protocol.

Run the c-treeACE monitor

Running the c-treeACE Monitor from a Windows Client will show the communication protocol used and the files opened:



The screenshot shows the c-treeACE Monitor application window. The title bar reads "c-treeACE Monitor (FAIRCOMS@192.168.26.136)". The menu bar includes "Server", "Options", "Tools", and "Help". Below the menu bar is a toolbar with icons for "LOCAL", "Start Snapshots", and a refresh icon. A tabbed interface shows "Active Connections" as the selected tab, with other tabs like "Files / Locks", "Files Stats", "Files History", "System Snapshot", "User Snapshot", "Sql Snapshot", "Snapshot Favorites", "System Configuration", "IO Performances", "Function Timing", "System Monitor", and "Memory Allocations". Below the tabs, there are radio buttons for "Show by Type" with options "All" (selected), "SQL", and "Isam", and a checkbox for "Show 'Internal' (Server Managed)". The main area displays a table of active connections.

Task #	User Name	Client IP Address	Node ID Info	Last Function	Active	Last Request Time	Last TRANBEG Time	Logon Time	Files	Memory	Comm Info
22	ADMIN (Current)	192.168.26.1	LAPTOP-DG0VLRP3	ctGetConnection...	yes	11/15/2022 12:11:08 PM	n.a.	11/15/2022 11:57:27 AM	0	59,352	F_TCPIP
21	GUEST	127.0.0.1		RTG_NEXTPREV	no	11/15/2022 11:56:42 AM	n.a.	11/15/2022 11:56:14 AM	8	331,880	FSHAREMM

Communication between the isCOBOL Server and the c-treeRTG Server with shared memory running on different containers

Performance Improvements

```
iscclient -c runtime.properties -hostname 192.168.26.136 IO_INDEXED
```

Working with shared memory gives you significantly better performance than working with TCP/IP. This table compares the performance results working with TCP/IP and with the Shared Memory between the docker containers.

IO_INDEXED NUM-TIMES=10000	TCP-IP	SHARED MEMORY
WRITE	4.40	0.85
READ	3.94	0.17
REWRITE	4.36	0.41
DELETE	3.35	0.58
TOTAL TIME	16.05	2.01

DOCUMENTATION HIGHLIGHTS

Options for Deploying Your Thin Client Application

There are different possibilities for the deployment of a thin client application.

While the server back-end is installed on your server, the thin client part can be distributed in different ways

depending on if you want a zero-client installation, if you prefer so use setups or if you're stuck to legacy technology like JWS (Java Web Start).

All these possibilities are explained in detail in the isCOBOL Documentation at:

[isCOBOL Application Server : Client deployment.](#)

It's Just COBOL, Not Mainframe Programming



Massimo Bertoli, President of Veryant, talks about the importance for a software house that uses open system COBOL to teach COBOL to new, young developers.

There's been a lot of talk recently about some COBOL developers aging out of the workforce and retiring, and how that should affect your future plans for your COBOL programs. However, an important difference has not been discussed – the mainframe COBOL programmer vs. the open system (Linux, Unix, Windows and Mac) COBOL programmer.

Most COBOL developers have extensive experience in areas that aren't necessary for open system COBOL programs; CICS and JCL are two such skills that have no relevance to your application. Open system COBOL is easier to develop and understand, and requires a different skillset than a Mainframe programmer.

If you've recently searched for a COBOL developer to hire, you've probably found that the available applicants are older, have experience in Mainframe technology that you don't need, and fall in the higher salary ranges. Instead, **why not teach COBOL to your existing developers?** They may know an object oriented programming language like Java or C++, and would view learning a procedural language like COBOL an interesting new experience.

These typically younger employees should find it easy to learn COBOL specific to the way it has been used in your application. For instance, they might spend some time learning embedded SQL if your application uses it. You can leverage your outgoing COBOL experts in an application-focused way, and shorten the learning time by eliminating unneeded lessons.

In fact, training current employees will save you time and money, provide your developers with

development and growth opportunities, and make your IT department more integrated, less siloed by language.

Another major benefit to teaching your existing programmers COBOL is the easy path to interoperability it offers. The future of COBOL typically does not involve new COBOL programming. Your COBOL is going to be just one piece of a multi-language application. Veryant is focused on this interoperability as a forward path for open systems COBOL.

“An important difference has not been discussed – the mainframe COBOL programmer vs. the open system COBOL programmer.”

Once your developers know COBOL, they'll be a great resource to add new features and integrate your COBOL application with new languages. isCOBOL already sets you on this path because it compiles and runs as a Java program. It's a natural fit to integrate with new Java development.

Mainframe programmers' skills may be difficult to learn, but just the COBOL language is all you need, and you probably already have the resources you need in your IT department. If your application uses some specific isCOBOL syntax (such as GUI, threads, or OOP) in addition to basic COBOL, Veryant is happy to either conduct the training, or provide supporting material for your internal training on these subjects. We're here to help you be successful.

COBOL

Tips and Tricks

One great way to modernize your programs is to change the color of the windows – for example from gray to white.

You probably already know you could use code-injection to do this to all of your screens at once from the properties file, without changing your code, using this property:

iscobol.compiler.gui.window.defaults=color rgb x#FFFFFF

BUT - if you already have the COLOR property set in your display window statement, this program setting will override any COLOR setting in your properties file, so code-injection won't work.

Here's another option: set gradient colors.

The gradient-color properties don't get overwritten by the program's COLOR setting. If you set the two gradient colors to the same color (for example, white), you can still use code-injection to change the window color without modifying your code.

For example you can use the following properties during when you compile the programs :

iscobol.compiler.gui.window.defaults=gradient-color-1 rgb x#FFFFFF gradient-color-2 rgb x#FFFFFF

Have You Seen This?



New YouTube Videos

[Starting the Application Server as a Windows service](#)

[ISMIGRATE from the Command Prompt](#)

[ISMIGRATE log files](#)

[c-treeRTG's Replication Manager - Memphis](#)

[WebClient Test Tool](#)

New Knowledge Base (KB) Articles

[Send SMS from isCOBOL using Twilio](#)



Evolution, without revolution



Contact Us

For supported customer email us at support@veryant.com

If you would like Veryant to contact you to schedule a technical product briefing, email us at info@veryant.com

If you would like Veryant to contact you for special quote or sales assistants email us at sales@veryant.com

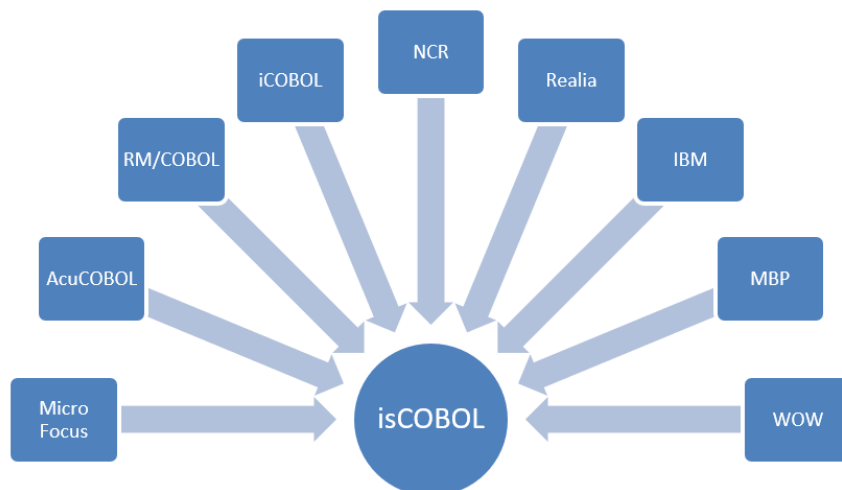
Corporate Headquarters

6390 Greenwich Dr., Suite 225
San Diego, CA 92122 - USA
Tel (English): +1 619 797 1323
Tel (Español): +1 619 453 0914

European Headquarters

Via Pirandello, 29
29121 - Piacenza - Italy
Tel: +39 0523 490770
Fax: +39 0523 480784
emea@veryant.com

As always, 2023R1 contains multiple compatibility additions – as we continue to make your conversion process as smooth, quick, and pain-free as possible.



veryant.com

Follow **Veryant** on



veryant.com

©2023 Veryant - All Rights Reserved