

# CODING WITH SOURCE STREAM FILES FOR INTEGRATED ENTERPRISE SOFTWARE MANAGEMENT

Modernize your IBM i development



## Contents

INTRODUCTION .....	3
THE STRUCTURE OF IBM i SOURCE CODE STORAGE IN SOURCE FILES .....	4
IBM i SYSTEM OBJECTS, CREATED OBJECTS AND COMPILED OBJECTS .....	4
THE INTEGRATED FILE SYSTEM (IFS) .....	5
STORING IBM i SOURCE CODE IN STREAM FILES IN YOUR OWN IFS DIRECTORIES .....	6
SOURCE STREAM FILE EDITING USING RATIONAL DEVELOPER FOR i (RDi) .....	6
Creating a QRPGLSRC member filter in the Remote Systems Explorer (RSE) Perspective .....	7
Creating an IFS filter in the Remote Systems Explorer (RSE) Perspective .....	8
Copying an RPGLE source member from a source file to a plain text file in an IFS location .....	11
COMPILING OBJECTS FROM SOURCE STREAM FILES .....	12
RECOMMENDATIONS .....	13
Using TURNOVER® Lifecycle Manager to manage it all .....	13
Managing enterprise source in a repository .....	14
CONCLUSION .....	15
ABOUT THE AUTHOR .....	16

## Introduction

Perhaps you're reading this because you're like me. You're an IBM i developer, analyst or operator with many years of experience doing things in a very specific way. You may have become accustomed to storing your IBM i source code in traditional IBM i libraries and source files. In addition, you may have used change management software to help your company control and audit software changes through development, QA and production environments.

On the IBM i, there is another way to store source for native object types such as programs, service programs and files, allowing it to be stored alongside source from other platforms in an enterprise-based repository system. Of course, I am talking about storing source code in stream files on the Integrated File System (IFS). This white paper will illustrate just how easy it is to start using stream files to store the source code for your IBM i objects.

Using IBM's Rational Developer for i (RDi) will allow you to easily edit and compile source code that is stored in stream files. Once you have made a start, you'll be ready to bring the IBM i source into a change management solution that enables you to manage it from development all the way through to production.

Finally, because stream files are plain text files, you can also share these items to wider enterprise source repositories such as Subversion and Git.

## The structure of IBM i source code storage in source files

One of the most intriguing things about the IBM i operating system (formerly OS/400) is the unique nature of object and source code storage. Most people who are trained as IBM i programmers and database developers will probably think first about IBM i objects such as \*PGM and \*FILE objects, among others. The operating system provides a unique storage structure consisting of libraries, which are simply named 'containers' for the aforementioned objects.

## IBM i system objects, created objects and compiled objects

Within the 'library' storage hierarchy of the IBM i operating system, some objects are system objects that are provided as part of the operating system and aren't meant to be modified.

Other objects can be created by users, with a simple operating system command, for example, to create a data area, use the operating system-supplied CRTDTAARA (Create Data Area) command. The information needed to create this object is contained within the command itself, and no other script or initialization information from any other source is needed to create it. All that is required are the parameters that are passed to the command, such as the length of the data area, whether it is character or decimal, and what the initial value is. The operating system already contains the necessary instructions to create the object.

On the other hand, many user-created objects must be 'compiled' from instructions written in a text area that is read by the compiling command, in order to create the object.

The IBM i has a special way of organizing and storing compile instructions used to create a specific object, that is, the concept of a member in a source file. This proprietary IBM i function allows you to store source instructions of a similar nature, such as RPGLE source, in a convenient and secure location, like recipes in a cookbook.

## The Integrated File System (IFS)

At the lowest level of operating system storage that is represented to the user, the IBM i storage configuration is accomplished via directories in a larger file system, called the Integrated File System (IFS). Within this file system, there is the root ('/') which represents the highest level of the file system. Under the root directory, there are other default directories that are provided by the IBM i operating system. These include:

- **/QOpenSys**  
UNIX-like objects and scripts
- **/QNTC**  
A 'network neighborhood' that establishes a link to Windows-based systems in your enterprise system network
- **/QSYS.LIB**  
The logical representation of the IBM i libraries that most users are accustomed to using, and the objects they contain

Beneath the root directory of the IFS, it is also possible to create your own named directories and subdirectories. These can be used to store text files of many kinds, including .js, .java, .xml, .py to name a few, along with compiled or containerized objects such as .class, .jar, .war and .zip file, and also binary objects such as .jpeg, .pdf, and many others.

## Storing IBM i source code in stream files in your own IFS directories

You can also create text files called 'stream files' in an IFS directory. These stream files can be used to store source code from traditional IBM i source files. One of the challenges facing IT departments these days is developing a cohesive, uniform strategy for maintaining and accessing source code from differing platforms, using modern tools to allow easy access and rapid deployment.

Consider storing your IBM i source members in stream files instead of in source members in traditional source files. In this way, you can create your own directory structures for development and testing environments. This is also a way to ultimately make these source items available to be stored to a modern repository such as Subversion or Git. You may wonder, is it just as easy to develop and maintain source code stored in IFS stream files, as it is to maintain them in traditional source members? Of course!

## Source stream file editing using Rational Developer for i (RDi)

Fortunately IBM provides an easy and modern tool for accessing and editing source code, not only for the IBM i but across your enterprise, namely, Rational Developer for i (RDi). Using the tools in RDi, you can browse or edit these members directly. RDi has the capability to edit source members in IBM i source files, and source code stored in stream files in the IFS.

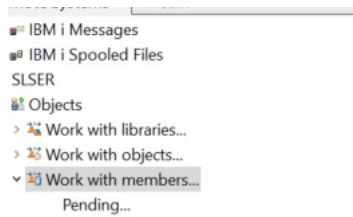
The unique nature of source physical files and members on IBM i does offer some positive features, such as maintaining source sequence numbers and 'last changed' dates on a per-row basis. When you move to source stream files those features are effectively lost, however, additional benefits can be realized, for instance, by implementing a more robust source version control system, such as Subversion or Git. These version control systems can maintain more detailed change history and allow recovery of old versions of the source code more readily.

Should you decide to copy your source 'members' into plain text files in an IFS directory, there is a very easy way to do this, using the Remote Systems Explorer perspective in RDi. To make this easy, you will need to create two filters; one which shows the QRPGLSRC file in your named library, and the other an IFS object filter, showing the directory where you intend to store your stream file source code. Once your filters are set up, you can copy an RPGLE source member from its IBM i source file directly into an IFS directory with just a couple of clicks. The new file on the IFS will be fully editable with RPGLE tools, using Rational Developer for i.

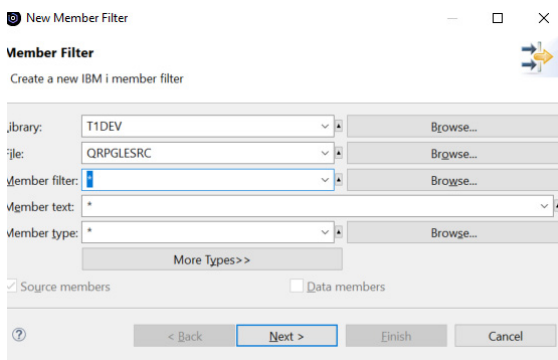
Creating filters is very easy, as illustrated in the following example.

### Creating a QRPGLSRC member filter in the Remote Systems Explorer (RSE) Perspective

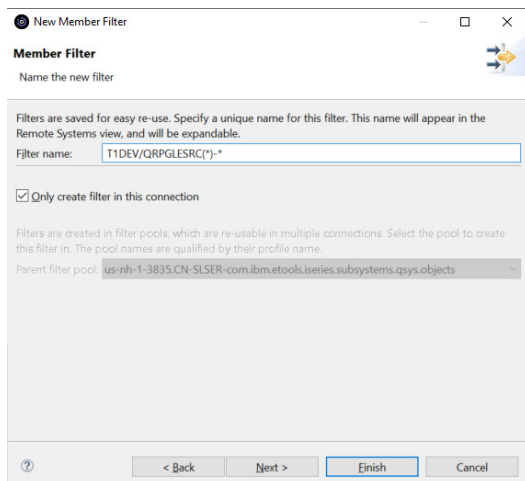
- Expand your system connection, then expand ‘Objects’
- Expand ‘Work with Members’



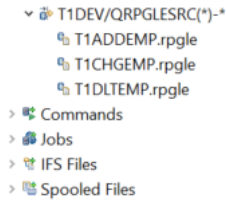
- The ‘New Member Filter’ pop-up window opens. Specify the library name, and QRPGLSRC for the file name. You can even filter further with member name wildcards, but for now, let’s just get the whole source file.



- Click ‘Next’. You can change the filter name, if you like, and you can also unclick the ‘Only create filter in this connection’ box if you have more than one IBM i system connection defined in RSE, and would like to create this filter across all systems.

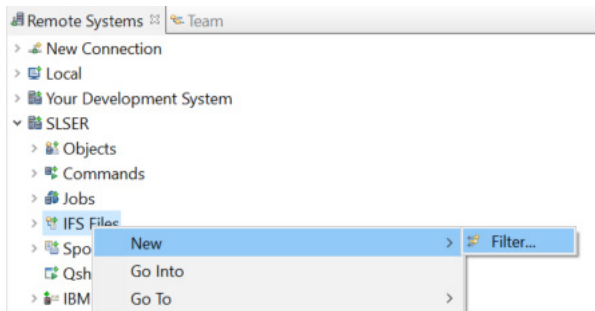


- **Click 'Finish'.** Now, you have a member filter which you can expand to see your RPGLE source members:

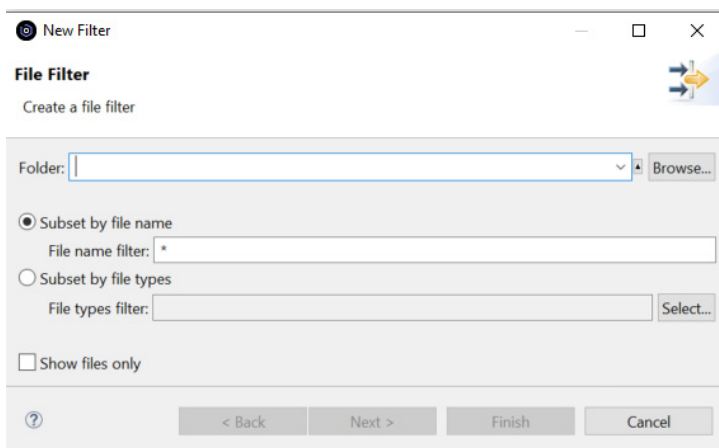


### Creating an IFS filter in the Remote Systems Explorer (RSE) Perspective

- **Expand your system connection, then right click on 'IFS files'. Select New > Filter...**

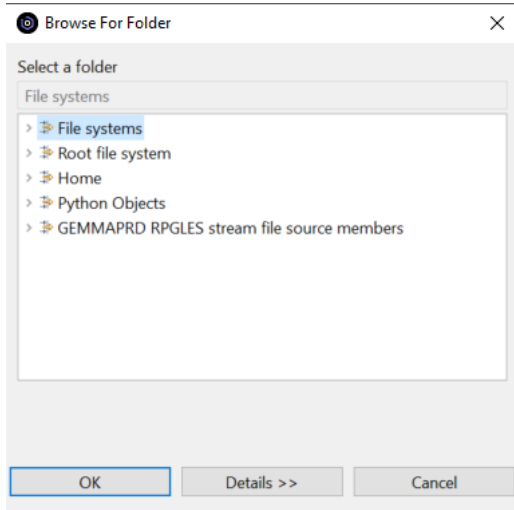


- **The 'New Filter' pop-up window opens. In the 'Folder' field, you can type in the IFS directory location if you know it, using forward slashes, or you can click the 'Browse' button.**

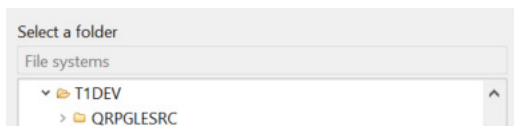




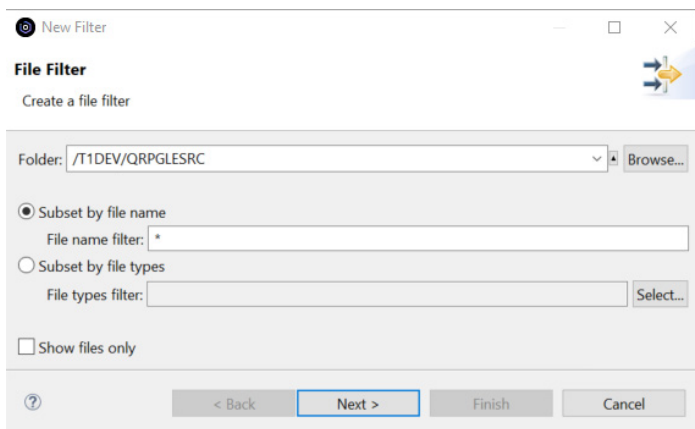
- If you click 'Browse', you will see another pop-up window.



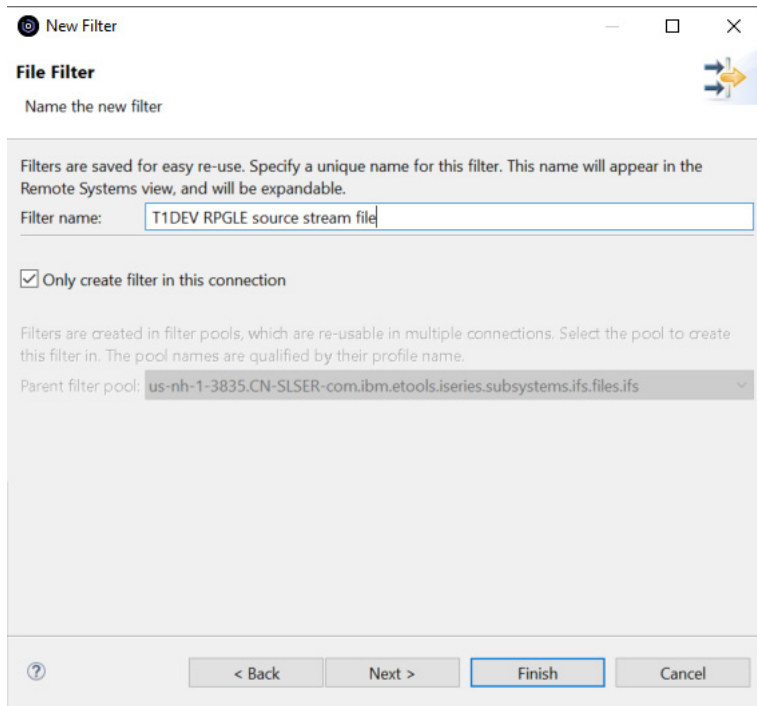
- To find your IFS directory, expand 'Root file system'. It may take a few moments to fully expand. Once the information is resolved and returns into the pop-up window, scroll down to locate your folder:



In this instance I have created a 'QRPGLESRC' folder in my development directory, which I will use to store the RPGLE stream files that I create here, or copy over from the QRPGLESRC source file in my T1DEV library. I have chosen to use 'QRPGLESRC' as the IFS folder name for consistency and clarity, but you can name the folder whatever you like.



- Click 'Next'. Here, you can provide a more descriptive filter name:



**New Filter**

**File Filter**

Name the new filter

Filters are saved for easy re-use. Specify a unique name for this filter. This name will appear in the Remote Systems view, and will be expandable.

Filter name:

Only create filter in this connection

Filters are created in filter pools, which are re-usable in multiple connections. Select the pool to create this filter in. The pool names are qualified by their profile name.

Parent filter pool:

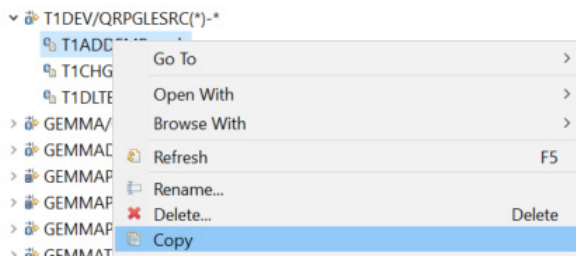
- Click 'Finish'.

You now have a new IFS filter available, when expanding 'IFS Files'.

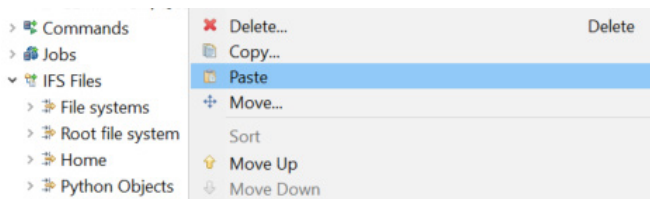
**Copying an RPGLE source member from a source file to a plain text file in an IFS location**

Once the filters are created, simply use the right-click context menus against your source member to copy it from your IBM i source file filter and to paste it into your IFS filter. An example is shown below:

- **Right-click to copy the specific source member from within the IBM i source file filter:**



- **Right-click on the IFS filter name to paste:**



You can even select multiple members from within the filter to copy at one time. You can also use the `CPYTOIMPF` command but our recommendation is to use Rational Developer for i as it provides a much richer environment for the developer to work in.

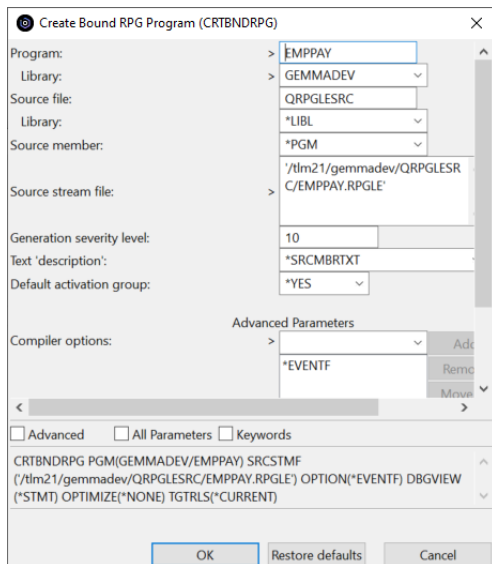
## Compiling objects from source stream files

The following commands at IBM i 7.4 can use stream files to create objects:

CRTBNDC	CRTCMOD	CRTSQLRPGI
CRTBNDCBL	CRTCBLMOD	CRTSRVPGM
CRTBNDCCL	CRTCLMOD	RUNSQLSTM
CRTBNDRPG	CRTRPGMOD	

It wasn't until v7.4 that stream file support was added for CLLE modules. In fact, even at v7.4, IBM i doesn't support the SRCSTMF parameter from commands like CRTPF and CRTLF, or from original program model (OPM) commands like CRTRPGPGM. A good commercial application lifecycle management solution will provide a way for you to store all of your source, compile physical and logical files, and even generate OPM programs, from stream files!

When compiling an object from a source stream file, you must use the stream file parameter (STMF) instead of the source member and library parameters, as shown in this CRTBNDRPG example:



You can see that the 'Source stream file' parameter specifies the IFS location of the RPGLE stream file in question which, in this case, is called EMPPAY.RPGLE. You will notice that with RDi, the 'Source file', 'Library' and 'Source member' names are filled in with default values, but those are ignored when the 'Source stream file' parameter (SRCSTMF) is used.

The RPGLE compiler will interpret the contents of the stream file just as it would a traditional source member.

## Recommendations

SoftLanding recommends that if you are new to using source stream files, that you focus on taking this approach for new projects, or for building new interfaces over existing systems, such as a new web front-end. It may be time-consuming and difficult to replace your existing code in a wholesale fashion for large software packages such as ERP.

For existing TURNOVER® Lifecycle Manager customers, SoftLanding's technical support department can help you get started with configuring your applications to support source stream files. Once you've established a few small source stream file items, you can build out from there.

### Using TURNOVER® Lifecycle Manager to manage it all

For many years, SoftLanding's market-leading application lifecycle management solution - TURNOVER® Lifecycle Manager - has provided the ability to manage changes made to 'native' source members and objects, that is, those that are contained within the library and source file structure, and which are used to compile objects defined within the IBM i operating system. The management of source changes is accomplished through the TURNOVER® Lifecycle Manager plug-ins for RDi. With these plug-ins, you can use TURNOVER® Lifecycle Manager's project and application management tools to fully audit and control all native software changes.

Now, TURNOVER® Lifecycle Manager also handles checkout, promotion, archiving and auditing for source code stored in stream files, as described in this white paper.

Once the new stream file has been created in the IFS, you can add the item to your TURNOVER® Lifecycle Manager worklist. Where the TURNOVER® Lifecycle Manager plug-ins use the built-in RDi editor, you will see that you can edit the new stream file directly, and you will see that the information in the source has been copied over in the exact position required (e.g. F specs at position 6, etc). The RDi stream file editor also provides prompting on the various specs, just as you would see when editing an IBM i source member, and also provides validation by showing any errors in the code.

Stream file objects are promoted and managed within TURNOVER® Lifecycle Manager's project management system and with worklists, just as all other object types are managed. As always, you have TURNOVER® Lifecycle Manager's powerful audit and promotion control features built in, to ensure more successful deployments and fewer headaches.

In addition to supporting compile commands that contain the SRCSTMF parameter, using TURNOVER® Lifecycle Manager, you can also store OPM RPG, CLLE and PF/LF source code on the IFS, where TURNOVER® Lifecycle Manager uses a special mechanism to convert the source into a temporary native member before compiling it. For OPM RPG, CLLE and PF/LF types, TURNOVER® Lifecycle Manager will archive and audit changes against source code stored in stream files, just as it does for all of the other object types.

Don't forget that the TURNOVER® Lifecycle Manager plug-ins for RDi can also manage the promotion and auditing of 'non-native' objects, as it has for many years. These refer to the other object 'extensions' referred to above, that can be contained within the directories and subdirectories of the IFS, such as .class, .jar, .xml, and .pdf, to name just a few.

### **Managing enterprise source in a repository**

Over the years, other methods have been developed to manage source code changes within an enterprise. The idea of a 'source repository', where code and associated artifacts can be managed and accessed by multiple developers at the same time, was first introduced with CVS and Subversion. These are centralized repositories where the information for the source versioning was kept on a central server. SoftLanding's TURNOVER® SVN is a Subversion-based example of a centralized repository.

In recent years, the Git framework has gained popularity and is now used by many enterprises. Git is a distributed repository, where the metadata and other source versioning information is duplicated on multiple servers and user systems. As you'd expect, stream files are just one example of source code that can be stored in a distributed repository. Keep in mind that Git is just one piece of the overall application lifecycle management puzzle, as you also need a good change management solution to gain visibility into code dependency, deployment of objects, and to manage defect tracking.

We will discuss the use of tools such as Git in managing IBM i source code in a future white paper.



# CONCLUSION

---

There are many factors that influence how a company decides to manage its software assets, particularly source originating from the IBM i, in addition to many other platforms. It is important to consider the visibility and positioning of IBM i software in the context of the wider enterprise. In this white paper, we have discussed the structure of traditional IBM i source and object management, from a structural, historical and operational perspective. We have explained the benefits of storing IBM i source code in stream files, and demonstrated the ease with which stream file source code can be created and managed using Rational Developer for i (RDi), using RDi's built-in filters.

TURNOVER® Lifecycle Manager is well-positioned to help you realize the most effective management of your software assets, with a well-established and transparent change process for native and stream file source code, in addition to 'non-native' IFS object types.

Finally, using a source code repository such as Subversion or Git, it is easy to store source items for many types of objects from many different platforms. This includes source stream files that you develop on the IBM i. Having the ability to manage all of this code alongside source created on other platforms aligns your IBM i development work with that of other teams and positions your IBM i assets and personnel to contribute strongly to the success of your organization.

## About the author

Leo Gemma is a seasoned IBM i software technician, with twenty years at SoftLanding. He has worked as both a technician and consultant, specializing in application lifecycle management and menu management. He is also an expert with TURNOVER® Lifecycle Manager's interfaces to tools such as CA 2E, LANSAs and ProGen.

Leo would love to hear your thoughts on this white paper and any other current issues related to the IBM i. He can be reached at [leog@softlanding.com](mailto:leog@softlanding.com).

## Why SoftLanding?

SoftLanding is a division of UNICOM® Global, specializing in application lifecycle management, problem and incident management, enterprise content management, automated operations, performance management, and menu management solutions for the IBM i, System i, iSeries and AS/400 platform.

SoftLanding's application lifecycle management solution defines and supports repeatable procedures for developing, deploying and maintaining IBM i, web and multi-platform applications, across the entire software development lifecycle.

The company's enterprise content management solution releases the power of digital communications through web, mobile, and email channels, without changing IT systems and applications.

SoftLanding's automated operations and performance management solutions keep core business systems running at optimum levels and prevent unplanned application downtime. Menu management solutions are also available, offering efficient, secure, flexible, and standardized access to corporate business applications running on IBM i.

SoftLanding's products and solutions are commercially available through UNICOM Global's UNICOM Systems and Macro 4 divisions.

**For more information on SoftLanding products and services visit [www.softlanding.com](http://www.softlanding.com).**

© Copyright 1990-2021 All Rights Reserved. Macro 4 Limited – a division of UNICOM Global.

UNICOM® Systems, Inc. UNICOM Plaza Suite 310, 15535 San Fernando Mission Blvd., Mission Hills, CA. 91345 USA

**Please contact us for more information:**



T: +1 818 838 0606  
E: [info@unicomsi.com](mailto:info@unicomsi.com)  
W: [www.softlanding.com](http://www.softlanding.com)



T: +44 (0)1293 872000  
E: [softlanding.uk@macro4.com](mailto:softlanding.uk@macro4.com)  
W: [www.macro4.com/softlanding](http://www.macro4.com/softlanding)