User Guide

ARCAD-Transformer RPG

Version 11.0.x

Publication Date: May, 2019

Prepared by the ARCAD Documentation Team

North America & LATAM
70 Main Street, Suite 203
Peterborough NH 03458
USA
1-603-371-9074
1-800-676-4709 (toll free)
sales-us@arcadsoftware.com

EMEA (HQ)
55 Rue Adrastée – Parc Altaïs
74650 Chavanod/Annecy
France
+33 450 578 396
sales-eu@arcadsoftware.com

Asia Pacific
c/o Pramex Intl Ltd
1 Austin Rd West Intl Commerce Centre
7107B 71/F Tsim Sha Tsui HONG KONG
Yau Ma Tei Hong Kong
sales-asia@arcadsoftware.com

Copyright © 1992-2019 by ARCAD. All rights reserved.

The following terms are names owned by International Business Machines Corporation in the United States, other countries, or both: AS/400®, ClearCase, ClearQuest®, DB2, DB2 Connect™, DB2 Universal Database™, ibm.com, IBM i, iSeries, System i, OS/400, Rational®, SP2, Service Pack, WebSphere. Java and all names based on Java are owned by Oracle Corp. in the United States, other countries, or both. Eclipse is a registered trademark of Eclipse Foundation, Inc. Other names of companies, products or services are the property of their respective owners.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Contact ARCAD

Headquartered in France at the foot of the Alps, ARCAD offers global services and has offices and partners all over the world. ARCAD partners with leading-edge companies throughout the world to offer full services, close to home.

Visit our website to Contact Us and find out more about our company and partners, or to request a demo.

The Customer Portal is intended for current and potential customers that have full or trial versions of ARCAD software. If you already use or are interested in using an ARCAD product, the portal lets you view all of your current licenses and generate your own temporary license keys for most ARCAD products. It grants you access to the ARCAD product knowledge base (FAQ, new releases, fixes, etc.) as well as the Release Notes and current documentation.

Do you have a request for change or have you encountered a bug? Log into the Helpdesk and create a ticket.

ARCAD guarantees consultant support 24 hours a day, 5 days a week (24/5) to registered members. Calls received are redirected, according to the hour, to put you in contact with a support team in or near your timezone.

Country	Address	Account Contact	Support Contact
France	ARCAD Software (HQ) 55 Rue Adrastée 74650 Chavanod	+33 4 50 57 83 96 sales-eu@arcadsoftware.com	support-eu@arcadsoftware.com
Germany	ARCAD Software Deutschland GmbH c/o Pramex International GmbH Im Trutz, Frankfurt 55 60322		
USA	ARCAD Software Inc. 70 Main Street, Suite 203 Peterborough, NH 03458	+1 (603) 371-9074 +1 (800) 676-4709 sales-us@arcadsoftware.com	support-us@arcadsoftware.com
Hong Kong	ARCAD Software Asia c/o Pramex Intl Ltd 1 Austin Rd West Intl Commerce Centre 7107B 71/F Kowloon Hong Kong	+852 3618 6118	support-asia@arcadsoftware.com
China	ARCAD Software Products Private Lmtd Room D07, floor 34, No.32, Zhujiang East Road Tianhe District, Guangzhou	+86 (020)22324643 +86 (020)22324649 sales-asia@arcadsoftware.com	
India	ARCAD Software D-280/281/282, Vibhuti Khand Gomti Nagar, Lucknow		

Table 1: Contact ARCAD

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Preface

Document purpose

This document is intended to guide you through configuring and using ARCAD-Transformer RPG.

Intended audience

This document is intended for all ARCAD-Transformer RPG users.

Related documentation

ARCAD technical documentation can be accessed from the product's online help or by logging into the Customer Portal on our website.

Related documentation
ARCAD Release Notes
ARCAD General Documentation
ARCAD-Glossary
ARCAD-Transformer RPG Installation Guide

Table 2: Related documentation

Publication Record

Unless stated otherwise, all content is valid for the most current version of ARCAD-Transformer RPG listed as well as every subsequent version.

Product version	Document version	Publication Date	Update record
≥ 11.0.x	3.0	May, 2019	Lock source member problem fixed. Converting Free to Fully-Free Form without any license control. Added free-form conversion options: Analyze Alpha to num. MOVE (ALPHTONUM) Keep indentation in the DS (KEEPDSIND) Source Line Date (SRCDATE)(not a new option but added to the preferences) Updated UI/icons, screenshots. Edited to remove incorrect instructions & vocabulary

Table 3: ARCAD-Transformer RPG User Guide publication record

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Product version	Document version	Publication Date	Update record
10.09.xx	2.9	June, 2018	Added *KEEP parameter to SRCDATE
10.08.xx	2.8	March, 2018	2 views added to manage conversion warnings: The Conversion Warnings view The Conversion Warning Resolution view Added Resolving conversion warnings. 10 new Free-Form conversion options: Convert declaration specs (CVTDCLSPEC) Empty Comment Lines (EMPTYCMT) CHECK (CHECKIND) SCAN (SCANIND) LOOKUP (LOOKUPIND) Z-ADD, Z-SUB (NUMTRUNCZ) ADD, SUB {Length(Fact1/Fact2)>Length(Result)} (NUMTRUNCA) ADD, SUB {Other} (NUMTRUNCB) MULT (NUMTRUNCM) DIV (NUMTRUNCD) New values for the Convert calculation specs (CVTCLCSPEC) parameter: *CHECK and *FREECHECK. Added pages to the conversion wizard, both for single-file and mass conversions. Added feature to copy item details. Added preference to force the usage of the plug-in as a stand-alone product.
10.07.xx	2.7	June, 2017	Added note about ARCAD_SMPL library ARCAD-Transformer RPG is now also known as *ARCAD RPG Converter for IBM i* when included in IBM's eConfig.
10.06.xx	2.6	May, 2016	Modified the CVT_CALL parameter to remove the possibility to prevent the replacement of CALL/CALLB with prototyped calls when the program is an array element variable. Content and document structure reorganized.
10.06.xx	2.5	February, 2016	Updated MAXNOTFREE instructions for "Fully-Free" sources. Revised description of how Conversion Units are generated and used.
10.06.xx	2.4	November, 2015	Added License Key Hold option to transfer license to another machine. Added parameters FULLYFREE, MAXNOTFREE and FIRSTCOL.
10.05.04	2.3	September, 2015	Moved the conversion engine installation and update instructions to the *Installation Guide*. Updated pagination.
10.05.00	2.1	June, 2015	Added multiple language option. Added parameters INDENTCMT, OPCODECASE, BLTFNCCASE and SPCWRDCASE.
10.04.11	2.0	May, 2015	Initial publication

Table 3: ARCAD-Transformer RPG User Guide publication record

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Contents

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

CONVERSION PROCESS

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

APPENDICES

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Tables

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Figures

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Introduction

1 About ARCAD-Transformer RPG

Prepare your RPG code for the new generation

ARCAD-Transformer RPG (aka *ARCAD RPG Converter for IBM i*) accelerates the conversion of your application to Free-Form coding. It is an optional module offered as an integral part of the ARCAD Pack for IBM i Modernization. ARCAD-Transformer RPG can also be purchased as a standard Eclipse IDE plug-in.

Reference

For more information about obtaining permanent licenses, Contact ARCAD.

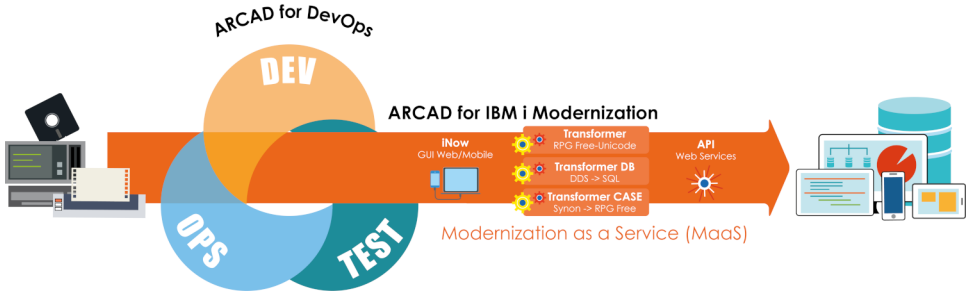


Figure 1: ARCAD-Transformer RPG in the ARCAD Group product suite

1.1 Business context

RPG IV has evolved into a modern business language, supporting procedures, data areas, data structures, additional data types and extended file support. Greater interoperability is offered between RPG and Java, XML and SQL. Also, RPG source code is far more readable thanks to Free-Form, blank lines, and comments.

Free-Form programs have the same source type and are compiled in exactly the same way as Fixed-Form RPG. The IBM RPG compiler allows the two styles to be mixed freely.

Experienced RPG developers can become proficient in RPG Free-Form with just a few days of learning. Free-Form brings not only the personal satisfaction of learning a new technology, but also modern language skills that can enhance your IT career for the future.

1.2 Functional constraints

Important!

Due to a new licensing format, starting from v10.07.00 ARCAD-Transformer RPG is only compatible with ≥v7.1 (see current documentation for exact version compatibility). If you are running an older IBM i OS or need a new license for an older version of the plug-in, you *must* upgrade.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

As part of Technology Refresh 7, a stand-alone PTF (number SI51094) for the RPG compiler is now available from IBM. This PTF enables you to compile Free-Form RPG and also manage the maximum number of conversions with ARCAD-Transformer RPG.

> **Note**
> If you specifically need to compile any SQLRPGLE sources that have been converted to Free-Form, you will then require the additional DB2 PTF group SF99701 level 26 (HyperPTF SF99701 in v7.1).

ARCAD-Transformer RPG is "DBCS capable": it is written 100% in Unicode and supports all CCSID sources, including Japanese, Chinese, and Korean.

Sources cannot be processed by the `ACVTRPGFRE` command if:

- they are (SQL)RPGLE source stored in IFS files (sources must be stored in a ***FILE** source file).
- the COPY clause contains C-specifications (their conversion is possible if they only contain declaration specifications.

For the lines surrounded by conditional compilation directives (`/IF` ... `/ENDIF`), the conversion is not guaranteed when the conditions are not effective during the compilation.

ARCAD-Transformer RPG converts only from RPG/400 (RPGIII), RPGIV to RPG Free-Form (and not older versions) because ARCAD uses the `CVTRPGSRC` IBM command, which converts from RPG/400 to RPGIV. Afterwords, ARCAD converts to RPG Free-Form using the `ACVTRPGFRE` Arcad command.

It's necessary to redesign RPGII before using ARCAD-Transformer RPG. RPGII programs use Primary/secondary Style F cards that are not supported by RPG free syntax. Also, they exclusively use I and O specs, which are not supported by RPG Free.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

2 Overview of the ARCAD-Transformer RPG perspective

The ARCAD-Transformer RPG perspective is a dedicated perspective used to manage conversion warnings, to manage and edit conversion lists and to execute massive conversions. This perspective consists of four elements.

2.1 The Conversion List view

This view displays the existing conversion lists, where each column represents one value of a list header. This view also has a toolbar to execute different actions such as adding or deleting a conversion list or refreshing the contents of the list. This view also allows you to open conversion lists for editing.

2.1.1 Filtering items in a conversion list

There are two ways to filter the items that are displayed in a conversion list. You can filter them by their conversion status or by their informational values.

Follow the subsequent steps to filter the contents of a list by the values found in the columns (**Library**, **Source File**, **Source Type**, **Object Type**, **Conversion Date** etc.).

Step 1 Right-click in the list area and select the **Filter...** option.

Step 2 Click **Add** to create a new filter in the **Filter** dialog.

Step 3 Select the status you want to use as a filter criteria from the drop down list in the **Columns** field.

Step 4 Confirm the **Operator type** by selecting the correct symbol from the drop-down list.

Step 5 Enter the **Search value**.

Follow the subsequent steps to filter the contents of a list by their conversion status.

Step 1 Right-click in the list area and select the **Filter** submenu.

Step 2 Check or uncheck the statuses as needed.

> **Note**
> **N/A** means that you want to display the conversion items which have not been converted.

2.2 The Conversion List editor

The **Conversion List** editor allows you to access the conversion items and all the functions that are applicable.

You can use this editor to:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- edit the values of the list header (connection name, target source file, etc).
- manage the contents of the list.
- assign an object type to the conversion item.
- execute a conversion.
- access the conversion status of each item.

2.3 The Conversion Warnings view

The **Conversion Warnings** view displays all the conversion warnings that have been issued on a member-by-member basis. The members displayed depend on the selected connection. If a member has not been checked for warnings yet or did not return warnings, it is not displayed.

This view provides an entry point to access the **Conversion Warning Resolution** view for a specific source member to update the conversion warnings issued.

It is also possible to browse for an original or a converted source member, to add members to a conversion list, to convert them or to check for warnings again from this view.

2.4 The Conversion Warning Resolution view

The **Conversion Warning Resolution** view displays all the individual conversion warnings, issued either for all the members that have been checked or for a specific member.

You can use this view to:

- view conversion warnings in more detail to know which part of a converted source's code is responsible for a conversion warning and how to fix it.
- update conversion warnings and indicate what should be done with each warning.
- browse for an original or a converted source member.
- add members to a conversion list.

2.5 Copying item details

ARCAD-Transformer RPG makes it possible to copy the value from a cell, or the entire line to easily retrieve all the details about a source member or a conversion warning. This feature is available in the following views:

- the **Conversion List** editor,
- the **Conversion Warnings** view, and
- the **Conversion Warning Resolution** view.

To copy the value of a specific cell, right-click on the cell from a list of source members or conversion warnings in one of the views, then select **Copy Text**. Paste the text in the location of your choosing.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

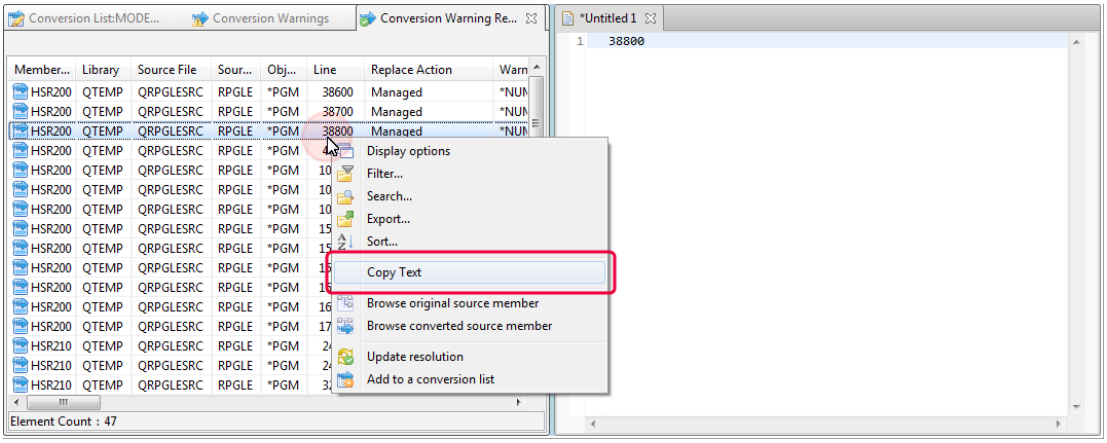


Figure 2: Copying the value of a specific cell

> **Note**
> In the screenshot above, the line number of the conversion warning is copied.

To copy an entire line with all the values, select an item from a list of source members or conversion warnings in one of the views, then press **CTRL+C**. Paste the text in the location of your choosing.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

3 About the conversion processes

Chapter summary

ARCAD-Transformer RPG provides many different choices of source code conversion. No one method is better than another. It is important to choose a standard conversion method for your company and to understand all of the options and their impact on the resulting modernized source code.

As part of the modernization process, a complete diagnostic compile listing is generated for each program being converted. This listing contains a field cross reference which ARCAD-Transformer RPG relies on in order to successfully convert the source code. This cross reference is an important component in the modernization process. In addition, the initial source code must be completely compilable. Without this, a new modernized source member will not be created.

> **Note**
> A diagnostic listing will not create a new program object and it will not replace any existing program objects.

There are four different steps in the conversion process.

> **Note**
> A member is an RPG source member if its source type is RPG, RPG38, RPT, RPT38 or SQLRPG.

3.1 The unitary conversion process

This process is executed to convert one source member selected from the **i Projects Navigator** or **Remote Systems** view.

- Check for conversion engine.
- IF the conversion engine is installed THEN check the availability of conversion units.
  - IF enough conversion units remain THEN execute The conversion process.
  - ELSE stop the process.
  - END.
- ELSE stop the process.
- END.

3.2 The conversion process

- IF the source member is an RPG source member THEN execute the The RPG to RPGLE conversion process.
  - IF the conversion failed THEN stop the process.
  - END

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- END
- Execute the The Free-Form conversion process.
  - IF the conversion failed THEN stop the process.
- END

3.3 The RPG to RPGLE conversion process

- Execute the standard `CVTRPGSRC` command using the preferences defined by the user.
- IF the conversion succeeds THEN use the newly converted RPGLE source member as the input to the Free-Form conversion.
- ELSE IF the conversion failed because the target source file does not exist THEN
  - IF the related preference has been checked THEN create the target source file and restart the conversion (this operation is executed only once).
  - END
- ELSE IF the conversion failed because the target source member already exists THEN
  - IF the related preference has been checked THEN remove the target source member and restart the conversion (this operation is executed only once).
  - END
- END

3.4 The Free-Form conversion process

- Select the parent library of the `ACVTRPGFRE` command according to the preference.
- Execute the `ACVTRPGFRE` command.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

4 About the conversion engine

The transformation of RPGLE to a more modern syntax may be inconvenient, but is easily considered an advantage.

In calculations and numerical assignments with Free syntax, it is no longer possible to have a result value larger than the capacity of the result variable, nor to have certain invalid values.

Where an old operation allowed you to truncate the result value (whether intended or not by the programmer!), the same operation performed with Free syntax generates an error. ARCAD also considers that this is a way to clean up code, because often these were hidden bugs.

Reference

For more information about installing and maintaining the conversion engine, refer to the *ARCAD-Transformer RPG Installation Guide*.

Example

Numeric Value Too Large (error at execution: MCH1210 Receiver value too small to hold result)

Before:

```
C     Move(p)  1234    WVar04       defined P(4,0)
C     Z-add    WVar04  WVar03       defined P(3,0)
     (result was 234 in WVar03)
```

After:

```
WVar04 = 1234;
WVar03 = WVar04;
```

Example

Invalid Numeric Value (error at execution: RNX0105: A character representation of a numeric value is in error)

Before:

```
C     Move     '45R'   WVarA3       defined A(3)
C     Move     WVarA3  WVar03       defined P(3,0)
    (result was -459 in WVar03)
```

After:

```
WVarA3 = '45R';
WVar03 = %Dec(%XLate(' ':'0':
          WVarA3):3:0);
```

4.1 About the ACVTRPGFRE command

The `ACVTRPGFRE` command is intended to be launched either natively or from RDi to prepare and/or perform the conversion of (SQL)RPGLE source to Free syntax.

This command, as a module of ARCAD-Transformer RPG, is licensed according to the number of conversions made (see Managing license keys on page 37).

Almost all of the H, F, D, P and C specifications can be converted to Free syntax. Regarding the declaration specifications (H, F, D, P), the conversion does not require the source to be compilable independently. The goal is to achieve the same for the calculation specifications (C). However, to do so,

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

it is important that members be written in or already converted to RPG III (RPG/400) *or higher* before carrying out the RPG Free transformation.

The purpose of this command is to convert ILE RPG to RPG Free. To do so, it carries out two separate actions:

1. the IBM command CVTRPGSRC, which converts any RPG III or IV to ILE RPG;
2. the ARCAD command ACVTRPGFRE, which converts any ILE RPG to RPG Free.

> **Important!**
> It is recommended to redesign any RPG II to RPG III or higher. If not, the modernization provided by ARCAD-Transformer RPG will not be possible.

It's easy to convert certain opcodes to Free syntax (e.g. EVAL, ADD, etc.), but others are much more difficult (e.g. MOVE, MOVEL, etc.). In fact, the instruction to generate very often depends on the type, length and dimensions of the fields used for factor 1, factor 2 or the result. This is why the command starts with a complete cross-reference (X-Ref) at the field level so it knows the characteristics of every field in the program.

> **Important!**
> This X-Ref calculation is based on a compilation (without creating the resulting object) which **must complete successfully**. It is therefore necessary to have libraries containing the sources and files used by the program online (see Preparing the environment on page 69).

Sometimes, compilation attributes are required or it may be necessary to execute commands prior to compilation (especially OVRDBF). In this case, it is recommended that you put these attributes and pre-compilation commands in the source to be analyzed as pre-compilation clause(s).

> **Note**
> All field declarations made in C-specifications are moved to D-specifications, unless the field is already declared elsewhere, in a file for example.

Because they can be distracting in the LPEX source editor, any special color attribute characters encountered in the comments in the changed lines are replaced by blanks.

If you define a source output member, it will contain the new converted source, but you cannot convert a source member inside itself.

> **Note**
> The resulting source is normally compilable, but you need to check the use of **%Found** and **%Equal** with the SCAN, CHECK, CHECKR and LOOKUP operation codes (see About operation codes on the next page).

You can also choose to only store the proposals for adding, modifying and deleting lines in the source modification file **AARFCHSF1**, keyed on CHS_CAPP = '*NONE', CHS_CENV = '*NONE', CHS_CVER = job number, CHS_JOBJ = object name, CHS_CTYPE = RPGLE or SQLRPGLE.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

4.2 Converting Free to Fully-Free Form

Due to a new licensing agreement, starting from v10.09.12 ARCAD-Transformer RPG can be used to convert previously-converted sources (RPGLE to Free) from their current Free-Form to Fully-Free.

Converting to Free-Form consists only in converting C spec to Free-Form. Converting to Fully-free, converts all specs (H, F, D, P) as well as C. If you have already converted to Free, you can now finish the conversion to fully-free without a (new) ARCAD-Transformer RPG license.

> **Note**
> When converting from Free (columns 8 to 80) to Fully-free, if your source includes comments in column 7, they will be converted from '*' to '//'.

When launching this conversion, the following parameters must be carefully defined.

Parameter	Required Value
Object Type (OBJTYPE)	*NONE
Convert calculation specs (CVTCLCSPEC)	*NO
Convert declaration specs (CVTDCLSPEC)	*NO
Convert to Fully-Free (FULLYFREE)	*YES
Max nbr of not free blocks (MAXNOTFREE)	*NONE

Table 4: Free to Fully-free required conversion parameters

> **Important!**
> If you launch the Free to Fully-free conversion using the ARCAD-Transformer RPG plug-in, these values are pre-defined and cannot be modified.

4.3 About operation codes

All RPG syntax revolves around operation codes. RPG is broken into declaratives and calculation instructions. Declaratives enable you to define: variables; access to files; the input/output parameters; and external prototypes (external programs called inside RPG programs). Calculation instructions are "managed" by an operation code word, with all its factors. The list contains strictly defined operation codes - each line in the list defines one opcode.

4.3.1 Opcodes that are never transformed

Only the following opcodes (rarely used) remain in classic syntax:

- MHHZO
- MHLZO
- MLHZO
- MLLZO

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

4.3.2 Opcodes that are only occasionally transformed

In most cases, the use of these opcodes can be converted, except in certain specific cases where there was no acceptable equivalent found in Free syntax:

- TIME: when the result field is defined with a length of 14.
- SCAN, CHECK, CHECKR: when the result field is an array.
- BITON, BITOFF: when factor 2 is a named-constant.
- POST: when the result field (DS name) is used.
- MOVE, MOVEL: when factor 2 or result field is a variable length field.
- CALL, PARM in some rare cases (see parameter Convert Program calls (CVT_CALL)).
- GOTO, TAG in some rare cases (see parameter Convert GoTo (CVT_GOTO)).
- KLIST, KFLD in some rare cases (see parameter Convert Key List (CVT_KLIST)).
- MOVEA in some rare cases (see parameter Convert MOVEA (CVT_MOVEA)).

4.3.3 Opcodes with error management using (e)

Many conventional operations codes allow use of the (e) extension to monitor execution errors, often followed by an %Error test. For many of these opcodes (e.g. CHAIN), there is the same possibility in Free with the same operation code.

But for the following opcodes, the Free syntax uses the `EVAL` operation which does not allow this extension:

- CHECK, CHECKR, SCAN
- OCCUR
- XLATE, SUBST
- ALLOC, REALLOC
- ADDDUR, SUBDUR, EXTRCT

In order to have the equivalent function in Free syntax, MONITOR provides the error handling, but a ruse has been used to successfully turn on/off the %Error indicator:

1. Two fields, **TTimeOk** and **TTimeError**, are created in the source.
2. The TEST instruction is used on these 2 fields to obtain the desired value for the %Error indicator.

Two comments containing ***CVTWNG** are also placed in the source to explain this process.

Example

Use of a Monitored ALLOC(e)

Before:

```
C       Alloc(e)  1000000       ptr01
C       If        %Error
C       ...
```

After:

```
// *CVTWNG : set %Error to '0'
  Test(e) TTimeOk;
  Monitor;
      ptr01 = %Alloc(1000000);
```


ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

```
          On-Error;
      // *CVTWNG : set %Error to '1'
            Test(et) *hms0 TTimeError;
        EndMon;
        If %Error;
        ...
```

4.3.4 Opcodes that manage %Found or %Equal

After execution, some conventional operation codes set the %Found or %Equal indicators that can then be tested.

For many of these opcodes (e.g. SETLL), there is the same possibility in Free syntax with the same opcode, but for the following opcodes, the Free syntax uses EVAL with a BIF %xxxxx(...) which no longer supports these two special indicators.

- CHECK, CHECKR (%Found)
- SCAN (%Found)
- LOOKUP (%Found & %Equal)

The management of the two indicators %Found and/or %Equal is no longer provided in the generated code. A comment containing ***CVTWNG** is added prompting you to verify if these two indicators are subsequently tested and to make any necessary corrections.

This comment is only added to your source if these indicators are used at least once in the program.

Example

SCAN, Followed by a %Found test

Before:

```
C      'B'       Scan     C01
C      If        %Found
C      ...
```

After:

```
If %Scan('B' : C01) > 0;
   // *CVTWNG : %Found is not updated by %Scan
EndIf;
If %Found;
...
```


ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

5 About the conversion operations

All of the opcodes included in the following tables were taken using the categories in the RPGLE reference guide. Each one has an explanation of the conversion used in Free.

Conversion operations	
Arithmetic operations	Indicator-setting operations
Array operations	Information operations
Bit operations	Initialization operations
Branching operations	Memory management operations
Call operations	Message operations
Compare operations	Move operations
Conversion operations	Move zone operations
Data-area operations	String operations
Date operations	Structured programming operations
Declarative operations	Subroutine operations
Error-handling operations	Test operations
File operations	XML operations

Table 5: Summary of conversion operations

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Arithmetic operations

If specified in positions 71-76, indicators are set after the converted instruction.

Operation code	Free version
ADD	Simple evaluation of the result value, with the + operator
DIV	Simple evaluation of the result value, with the / operator
MULT	Simple evaluation of the result value, with the * operator **Note** Use of the MULT operation code with one of the following values in factor 1 or 2: 100.0001, 10000.01 or 10000.0001. If the variables on the line have a length of 6 (or 8 for the last case), it is assumed to be a method/technique used to invert the format of a date between MmDdYy and YyMmDd (or MmDdYyyy and YyyyMmDd in the last case). For these cases, an equivalent process is done, with Eval, %Dec, %Subst and %EditC.
MVR	Evaluation with %Rem(...), but placed before the previous DIV instruction. **Note** There is a risk if the result variables of the DIV or MVR are also the DIV operands.
SQRT	Simple evaluation of the result value, with %Sqrt(...)
SUB	Simple evaluation of the result value, with the - operator
Z-ADD	Simple numeric assignment
Z-SUB	Simple numeric assignment with negative value

Table 6: Arithmetic operations conversion

Array operations

Operation code	Free version
LOOKUP	Evaluation using %LookUp, %LookUpXX (or %TLookUp, %TlookUpXX) and set result indicator *INxx if required. In the case of an array with a variable index, the index is set to 1 for a failed lookup. **Note** Addition of a comment warning: *CVTWNG: %Equal & %Found are not updated by %LookupXX
MOVEA	See parameter Convert MOVEA (CVT_MOVEA).
SORTA	Simple conversion to Free syntax for an opcode already using extended factor 2.
XFOOT	Evaluation of %XFoot. If specified in positions 71-76, indicators are set afterwards.

Table 7: Array operations conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Bit operations

Operation code	Free version
BITOFF	Evaluation using a combination of %BitAnd and %BitNot. No conversion if factor 2 is a named constant.
BITON	Evaluation using a combination of %BitOr. No conversion if factor 2 is a named constant.
TESTB	Separate evaluation for each resulting indicator by comparing with a hex value (sometimes with %BitAnd).

Table 8: Bit operations conversion

Branching operations

Operation code	Free version
CABxx, GOTO, TAG	See parameter Convert GoTo (CVT_GOTO).
ITER, LEAVE	Same opcode without parameters in Free syntax. **Note** This can sometimes be replaced by "ATag = '*LEAVE' or '*ITER'", for the management of GOTO (backwards) transformed to structured programming.
LEAVESR	Same opcode without parameters in Free syntax.

Table 9: Branching operations conversion

Call operations

Operation code	Free version
CALL, CALLB, PARM, PLIST	See parameter Convert Program calls (CVT_CALL).
CALLP, RETURN	Simple conversion to Free syntax for opcodes already using extended factor 2.

Table 10: Call operations conversion

Compare operations

Operation code	Free version
ANDxx, ORxx	Addition of AND or OR then a test of factor 1 and 2, without terminating the instruction if followed by another ANDxx or ORxx.
CABxx	If there are indicators, first test factor 1 and factor 2, then manage branch condition. See parameter Convert GoTo (CVT_GOTO).
CASxx	If there are indicators, first test factor 1 and factor 2, then manage subroutine execution according to the condition.

Table 11: Compare operations conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Operation code	Free version
	Note However, the conversion of the CASxx, CAS, ENDCS group has 2 possibilities: • If at least one of the CASxx statements has an indicator in positions 71-76, it will be converted to If ..., Exsr ..., EndIf. • If none of the CASxx statements has an indicator in positions 71-76, this will be structured better: Select, When ... , Exsr ..., ... EndSl.
COMP	Setting of each indicator, comparing factor 1 and factor 2.
DOU, DOW, IF, WHEN	Simple conversion to Free syntax for opcodes already using extended factor 2.
DOUxx, DOWxx, IFxx, WHENxx	Use of the opcodes DoU, DoW, If or When with a test of both factor 1 and 2, without terminating the instruction if followed by ANDxx or ORxx.

Table 11: Compare operations conversion

Conversion operations

Operation code	Free version
MOVE, MOVEL	See Move operations.

Table 12: Conversion operations conversion

Data-area operations

Operation code	Free version
IN, OUT, UNLOCK	Equivalent code in Free, with additional management of the error indicator if specified in positions 73-74.

Table 13: Data-Area operations conversion

Date operations

Note
Manage the operation code extender (e) with Monitor/On-Error/EndMon.

Operation code	Free version
ADDDUR	Evaluation of the new Date, Time, TimeStamp fields using %Years, %Months, ... %Hours, etc.
EXTRCT	Evaluation of Date, Time, TimeStamp fields using %Subdt; conversion to alpha with %EditC if necessary.
MOVE, MOVEL	When factor 2 and/or the result field contain a Date/Time/TimeStamp field,

Table 14: Date operations conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Operation code	Free version
	conversion using %Date, %Time, %TimeStamp, or %Char, %Dec using the date/time format of the field. See Move operations.
SUBDUR	Evaluation according to the operation and the field types for factors 1 and 2 and the result field: • either using %Diff • or by subtracting a time/duration using %Years, %Months, ... %Hours, etc.

Table 14: Date operations conversion

Declarative operations

Operation code	Free version
*DTAARA DEFINE	Transfer of the associated field declaration to a *DTAARA in D-specs, adding the DTAARA(xxxx) keyword. **Note** Not done if this declaration is in a COPY clause.
KFLD, KLIST	See parameter Convert Key List (CVT_KLIST).
*LIKE DEFINE	Transfer of the field declaration to D-specs.
PARM, PLIST	See parameter Convert Program calls (CVT_CALL).
TAG	See parameter Convert GoTo (CVT_GOTO).

Table 15: Declarative operations conversion

Error-handling operations

Operation code	Free version
MONITOR, ON-ERROR, ENDMON	Simple conversion to Free syntax for opcodes already using extended factor 2.

Table 16: Error-handling operations conversion

File operations

Operation code	Free version
ACQ, CHAIN, CLOSE, COMMIT, DELETE, EXCEPT, EXFMT, FEOD, FORCE, NEXT, OPEN, READ, READC, READE, READPE, REL,	The equivalent operation code exists in Free for all these file operations with 0, 1 or more parameters. Followed possibly by setting indicators *INxx, if they were specified in positions 71-76, testing the %Error, %Found or %Equal indicators. ***Reference*** See the processing of key lists (Convert Key List (CVT_KLIST)).

Table 17: File operations conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Operation code	Free version
ROLBK, SETGT, SETLL, UNLOCK, UPDATE, WRITE	
POST	As above, but not converted when the result field is used - no equivalent in Free syntax.

Table 17: File operations conversion

Indicator-setting operations

Operation code	Free version
SETOFF, SETON	Evaluation to 0 or 1 for 1, 2 or 3 indicators, successively.

Table 18: Indicator-setting operations conversion

Information operations

Operation code	Free version
DUMP	Equivalent opcode in Free.
SHTDN	Indicator evaluation *INxx with %ShtDn.
TIME	Four different cases, according to the type/length of the result field: 1. Field type Date/Time/TimeStamp: Evaluation with %Date, %Time, %TimeStamp 2. Numeric field of 6,0: Evaluation with %Dec(%Time) 3. Numeric field of 12,0: Evaluation with %Dec(%Time) * 1000000 + %Dec (%Date:*JOBRUN) (This would not be OK if the format of the job date was *JUL, but this is without doubt never used). 4. Numeric field of 14,0: Stays in traditional format, because it is impossible to have a format with the correct mapping of the century according to the format of the job.

Table 19: Information operations conversion

Initialization operations

Operation code	Free version
CLEAR, RESET	Equivalent opcode that exists in Free, with 1 or 2 parameters.

Table 20: Initialization operations conversion

Memory management operations

Operation code	Free version
ALLOC,	Evaluation of a pointer with %Alloc or %ReAlloc.

Table 21: Memory management operation conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Operation code	Free version
REALLOC	**Note** Manage the operation code extender (e) with Monitor/On-Error/EndMon.
DEALLOC	Equivalent opcode that exists in Free, with 1 parameter; then setting of indicator *INxx, if specified as the error indicator.

Table 21: Memory management operation conversion

Message operations

Operation code	Free version
DSPLY	Equivalent opcode that exists in Free, with 1, 2 or 3 parameters.

Table 22: Message operation conversion

Move operations

Operation code	Free version
MOVE, MOVEL	This operation code, widely used in traditional syntax, performs operations for which the behavior depends on the type of the variables and their length; all of the following cases are covered: Figurative constant in factor 2 (*Blank, *Zero, *Hival, *Loval, *ALL'o', *ALL'xxxx'). • With or without (p) as operation extender • Variable or fixed length field • Factor 2 field with a length less than the result field • Factor 2 field with a length greater than or equal to the result field • Assignment with numeric conversion <-> alpha using %XLate, %Dec, %EditC, and possibly digital shifting by multiplying or dividing by 10, 100, 1000 etc... • When factor 2 and/or the result field contain a Date/Time/TimeStamp field, conversion using %Date, %Time, %TimeStamp, or %Char, %Dec using the date/time format of the field. If specified in positions 71-76, indicators are set after the converted instruction. **Note** The assignment between a date/time or numeric field and a variable-length field is not converted.
MOVEA	See parameter Convert MOVEA (CVT_MOVEA).

Table 23: Move operation conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Move zone operations

Operation code	Free version
MHHZO, MHLZO, MLHZO, MLLZO	These opcodes (rarely used) have no equivalent in Free syntax - they always stay in traditional syntax.

Table 24: Move zone operation conversion

String operations

> **Note**
> For CHECK, CHECKR, SUBST, XLATE, manage the operation code extender (e) (or an error indicator in positions 73-74) with Monitor/On-Error/EndMon.

Operation code	Free version
CAT	About 20 different cases (fairly simple) generating a block of 1 to 20 lines, related to the following variations: • with or without (p) as operation extender • Number of blanks not specified in factor 2 • Number of blanks is 0 in factor 2 (CAT Var:0) • Number of blanks between 0 and 5 in factor 2 (CAT Var:2) • Number of blanks greater than 5 or variable in factor 2 (..CAT Var:Var2..) In certain cases, a global variable NCatLen is added to allow correct management of concatenations.
CHECK, CHECKR	Use of %Check or %CheckR. Addition of a comment warning: *// *CVTWNG : %Found is not updated by %Check (r).* Not converted when the result field is an array.
SCAN	Use of %Scan. Addition of a comment warning: *// *CVTWNG : %Found is not updated by %Scan.* Not converted when the result field is an array.
SUBST	Evaluation using %Subst with, if necessary, %Subst for the result variable if there is no operation extender "(p)".
XLATE	Evaluation using %XLate with, if necessary, %Subst for the result variable if there is no operation extender (p).

Table 25: String operations conversion

Structured programming operations

Operation code	Free version
ANDxx, ORxx	Addition of AND or OR then a test of factor 1 and 2, without terminating the instruction if followed by another ANDxx or ORxx.

Table 26: Structured programming operations conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Operation code	Free version
DO	Several possible cases: 1. DO or DO 1 are replaced by **DoU '1'**. 2. DO *HIVAL is replaced by **DoW '1'**. 3. Other cases of DO with result field: replaced by **For ...**, with the result variable as an index. 4. Other cases of DO without result fields: replaced by **For NForIdxNNNN = ...**, where NForIdxNNNN is a local variable created (with NNNN = 0001, 0002, ...) for each loop managed. If an index is specified on the corresponding ENDDO (or END) statement, then this is set to **By xxxx**, following the **For ...**
DOU, DOW, IF, FOR, WHEN	Simple conversion to Free syntax for opcodes already using extended factor 2.
DOUxx, DOWxx, IFxx, WHENxx	Use of the opcodes DoU, DoW, If or When with a test of both factor 1 and 2, without terminating the instruction if followed by ANDxx or ORxx.
ELSE, ELSEIF, SELECT, OTHER	Same opcode exists in Free syntax.
END, ENDxx	The opcode END is always replaced by EndIf, EndDo, EndFor, EndMon or EndSl according to the instruction at the start of the group to which it relates. The opcode ENDDO may become EndFor.
ITER, LEAVE	Same opcode without parameters in Free syntax. This can sometimes be replaced by ATag = '*LEAVE' or '*ITER', for the management of GOTO (backwards) passed to structured programming.

Table 26: Structured programming operations conversion

Subroutine operations

Operation code	Free version
BEGSR, ENDSR, EXSR, LEAVESR	Same opcode in Free syntax. ***Reference*** For ENDSR with a label in factor 1, see parameter Convert GoTo (CVT_GOTO). To convert the subroutines to ILE procedures, see parameter Convert Subr. to procedures (CVT_SUBR).
CASxx, CAS, ENDCS	If there are indicators, first test factor 1 and factor 2, then manage subroutine execution according to the condition. However, the conversion of the CASxx, CAS, ENDCS group has 2 possibilities: • If at least one of the CASxx statements has an indicator in positions 71-76, it will be converted to If ..., Exsr ..., EndIf. • If none of the CASxx statements has an indicator in positions 71-76, this will be structured better: Select, When ..., Exsr ..., ... EndSl.

Table 27: Subroutine operation conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Test operations

Operation code	Free version
TEST	Use of the same opcode in Free syntax: Test(e)... Then setting of indicator *INxx, if specified in positions 73-74.
TESTB	Evaluation of each resulting indicator, by comparing with a hex value (sometimes with %BitAnd).
TESTN	Evaluation of indicator(s) specified in positions 71-76, more or less complex: with %Checkr('0123456789'...), %Check(X'C0D0F0') or %BitAnd, ... in order to ensure a similar behavior. A comment in the form *// Test xxx : numeric or blank ?* is also added to explain each test.
TESTZ	Evaluation of indicator(s) specified in positions 71-76, more or less complex: with %BitAnd, and comparison to allowed hex values (X'C0', X'D0', X'50', X'60').

Table 28: Test operation conversion

XML operations

Operation code	Free version
XML-SAX, XML-INTO	Same opcode in Free syntax.

Table 29: XML operation conversion

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Configuration

6 Introduction to configuration

Before using ARCAD-Transformer RPG it is necessary to configure the parameters and register your activation key. This section describes the different parameters found in ARCAD-Transformer RPG preferences as well as how to manage the product's licenses.

> ***Reference***
> For more information about installing and upgrading, refer to the *ARCAD-Transformer RPG Installation Guide*.

Follow the subsequent steps to access the various preference pages.

Step 1 Open the RDi **Preferences** (*Window* > *Preferences*).

Step 2 Expand the ARCAD-Transformer RPG category.

By clicking directly on the **ARCAD-Transformer RPG** node in the **Preferences** window, you get access to the **Force the usage of the stand-alone product** preference:

- When this preference is checked, the ARCAD-Transformer RPG plug-in is used as a stand-alone and the standard **Available IBM i Connections** dialog will be displayed to select a connection.
- Leave this box unchecked to use the ARCAD-Transformer RPG plug-in as any other plug-in in the ARCAD suite, with the typical **Arcad Connection** dialog.

To change the preference, check or uncheck the **Force the usage of the stand-alone product** box, then click **Apply** to save the modification.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

7 Managing license keys

> **Important!**
> The conversion engine must be installed before registering an activation key.

The number of transformations you can make using ARCAD-Transformer RPG is based on conversion units. One unit is consumed each time a source member is successfully converted, regardless of its size and regardless of how many times it has already been converted.

> ***Example***
> Converting one source costs one conversion unit. Each successive conversion of the same source also costs one conversion unit.

The conversion engine counts conversion units based on your license. You can call the conversion engine until all the available units in your license are consumed. To continue using the engine after all of your units are consumed, request and activate a new license for ARCAD-Transformer RPG. Licenses are managed by activation keys.

Temporary activation keys

Temporary activation keys for ARCAD-Transformer RPG enable you to evaluate the product for free by activating 10 conversion units. The temporary ARCAD-Transformer RPG activation key is sent to you by email, following a successful download from the website. This temporary key is valid on the machine specified (defined by its serial number and LPAR number) for 15 days following the download. It is recommended to activate the key during this period. If you do not activate the key in this period, please contact ARCAD support or the sales team who can re-generate the temporary key for you, granting you an overall total of 10 conversions.

Once the temporary key is active, it is valid for one year. Please note that successive conversions of the same source member do consume conversion units. For a given serial number and LPAR, a used temporary key cannot be extended or re-issued. If you wish to continue to use the product after the 10 free conversions are made, please contact your local ARCAD sales team to purchase a permanent license.

Permanent activation keys

Once purchased, a permanent activation key allows you to use the conversion unit you have paid for, with no expiration date.

You can also contact your local IBM representative with the following product ID: 5725-L13 and part number: D12EWLL.

7.1 Activating keys

Activation keys are managed from the plug-in's **Activation Key** preferences [*Window* > *Preferences* > *ARCAD-Transformer RPG* > *Activation Key*].

ARCAD-Transformer RPG can be installed on several different servers. Select the IBM i connection related to the server you want to manage.

Follow the subsequent steps to select an IBM i connection.

Step 1 Click the Browse icon to the right of the **Connection** field.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Step 2 Select the appropriate connection in the **Available IBM i Connections** dialog.

Step 3 Click **OK**.

Result The connection is automatic and if successful, the **License Status** displays.

Permanent Activation Key	The current, active license is permanent.
Temporary Activation Key	The current, active license is temporary.
No Valid Activation Key	No active license is found.

Table 30: License Status

7.1.1 Viewing activation key details

Before registering (activating) a key, you can check its contents to be sure it is really what you want to activate. Follow the subsequent steps to view activation key information for ARCAD-Transformer RPG.

Step 1 Enter the key you wish to check in the **Activation Key** field.

Step 2 Click **Key Info**.

Result All information concerning the key is displayed, including the total number of conversion units in the license, the number used and the number of conversions you can still make.

7.1.2 Registering an activation key

A key must be registered (activated) for your product before you can use it. Follow the subsequent steps to register an activation key.

Step 1 Enter the key you wish to activate in the **Activation Key** field.

Step 2 Click **Activate**.

Result All information concerning the key is displayed, including the total number of conversion units in the license, the number used and the number of conversions you can still make.

Due to a new licensing agreement, starting from v10.09.12 ARCAD-Transformer RPG can be used to convert previously-converted sources (RPGLE to Free) from their current Free-Form to Fully-Free.

Converting to Free-Form consists only in converting C spec to Free-Form. Converting to Fully-free, converts all specs (H, F, D, P) as well as C. If you have already converted to Free, you can now finish the conversion to fully-free without a (new) ARCAD-Transformer RPG license.

> **Note**
> When converting from Free (columns 8 to 80) to Fully-free, if your source includes comments in column 7, they will be converted from '*' to '//'.

When launching this conversion, the following parameters must be carefully defined.

Parameter	Required Value
Object Type (OBJTYPE)	*NONE
Convert calculation specs (CVTCLCSPEC)	*NO

Table 31: Free to Fully-free required conversion parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Required Value
Convert declaration specs (CVTDCLSPEC)	*NO
Convert to Fully-Free (FULLYFREE)	*YES
Max nbr of not free blocks (MAXNOTFREE)	*NONE

Table 31: Free to Fully-free required conversion parameters

Important!
If you launch the Free to Fully-free conversion using the ARCAD-Transformer RPG plug-in, these values are pre-defined and cannot be modified.

7.2 Transferring keys to different machines

You can transfer your ARCAD-Transformer RPG license between two machines in order to use the remaining conversion units available on a license key on a second IBM i system. Transferring your license key involves contacting your ARCAD Software Provider and entering the information shared with you in the **Preferences** menu.

The idea is to disable a license on one machine (hold it) in order to receive a new activation key for another machine.

Follow the subsequent steps to hold your current key for an IBM i connection and make the associated conversion units available to use on a different machine.

Step 1 Contact your ARCAD Software Provider. Ensure that the person you are contacting is able to generate Hold Keys, like the support team.

In your message, give details concerning your request and provide your current machine's serial number and LPAR.

Result Your provider will send you the hold key that corresponds to your request.

Step 2 Open the **Hold ARCAD Transformer RPG key** preference page.

Step 3 Connect to the current IBM i for which you want to deactivate the license in the **Preferences** menu.

Click the Browse icon to the right of the **Connection** field. Select the appropriate connection in the **Available IBM i Connections** dialog.

Click **OK**.

Step 4 Enter the hold key that you received in Step 1 in the first area.

Step 5 Click the **Hold** button.

Result A special deactivation key is generated in the second area which includes the number of conversion units still available in the initial license.

Step 6 Send the deactivation key to your ARCAD Software Provider.

This new key *must* be sent back to your software provider. They will use this deactivation key to analyze the number of conversion units your license has and create a new activation key for your second system that contains the same amount of units.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Result Your provider will send you the activation key that corresponds to your request.

Step 7 Register the new activation key for the new system (connection).

Reference

For more information about registering keys, refer to Registering an activation key on page 38.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

8 Changing the language

ARCAD-Transformer RPG is available in the following languages: English, French, German and Japanese.

To manage the language in which the messages and command prompts for ARCAD-Transformer RPG are displayed, open the **Change Language for IBM i product** preference page.

> **Note**
> Only one language can be selected at a time and the option is only available when ARCAD-Transformer RPG is used without ARCAD (Library ARCAD_RPG). You cannot change the language when using an ARCAD Server.
>
> Some messages or texts may not be translated and therefore are only displayed in English.

Step 1 To change the language for your system, select an IBM i connection.

The product can be installed on several different servers. Select the IBM i server to manage.

1. Click the Browse icon to the right of the **Connection** field.
2. Select the appropriate connection in the **Available IBM i Connections** dialog.
3. Click **OK**.

Step 2 Select the language from the **Available languages** drop-down list.

Step 3 Click **Apply**.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

9 Managing the Free-Form conversion options

The default status of the Free-Form conversion options are managed from the plug-in's preferences. These parameters enable you to predefine a number of transformation preferences.

When transforming a source member, the same parameters are available to change as needed in the **Converted Source Member Properties** window. However, configuring them in the preferences menu defines the default statuses.

The following table contains a link to a complete description of each parameter and the values allowed for each parameter.

> **Note**
> Conversion parameters are all optional. If no value is entered or selected, the following default values (**in bold**) are used.

Group	Conversion option	Values
	Replace Existing Member (REPLACE)	YES, **NO**
	Convert declaration specs (CVTDCLSPEC)	**YES**, NO
	Convert to Fully-Free (FULLYFREE)	YES, **NO**
	Max nbr of not free blocks (MAXNOTFREE)	***NONE**, 1-999, *NOMAX
	First Column (Fully-Free) (FIRSTCOL)	**1**, 2-5
	Convert Program calls (CVT_CALL)	**YES**, NO
	Convert GoTo (CVT_GOTO)	*NO, *BASE, ***ADVANCED**
	GOTO Label (TAGFLDNAME)	*Character value*, **ATag**
	Convert Key List (CVT_KLIST)	**YES**, NO
	Convert MOVEA (CVT_MOVEA)	*NO, *BASE, ***ADVANCED**
	Convert Subr. to procedures (CVT_SUBR)	YES, **NO**
	Use %ParmNum (USEPARMNUM)	**YES**, NO
	Indentation Size (char) (INDENT)	0-5, **2**
	Indent Comments (INDENTCMT)	**YES**, NO
	Empty Comment Lines (EMPTYCMT)	***KEEP**, *BLANK, *ONELINE, *REMOVE
	Keep indentation in the DS (KEEPDSIND)	*YES, ***NO**, *WNG1,
	Case for operation codes (OPCODECASE)	***MIXED**, *UPPER, *LOWER
	Case for the B.i.F. (BLTFNCCASE)	***MIXED**, *UPPER, *LOWER
	Case for special words (SPCWRDCASE)	***MIXED**, *UPPER, *LOWER
	Case for key words (KEYWRDCASE)	***MIXED**, *UPPER, *LOWER
Analyze Indicator Problems	CHECK (CHECKIND)	***WNG1**, *YES, *NO
	SCAN (SCANIND)	***WNG1**, *YES, *NO
	LOOKUP (LOOKUPIND)	***WNG1**, *YES, *NO
Analyze Numeric Truncation	Z-ADD, Z-SUB (NUMTRUNCZ)	***WNG1**, *WNG2, *YES, *NO
	ADD, SUB {Length(Fact1/Fact2)>Length	***WNG1**, *WNG2, *YES, *NO

Table 32: Summary of conversion options

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Group	Conversion option	Values
	(Result)} (NUMTRUNCA)	
	ADD, SUB {Other} (NUMTRUNCB)	*WNG1, *WNG2, *YES, ***NO**
	MULT (NUMTRUNCM)	***WNG1**, *WNG2, *YES, *NO
	DIV (NUMTRUNCD)	***WNG1**, *WNG2, *YES, *NO
	Analyze Alpha to num. MOVE (ALPHTONUM)	***YES**, *NO
	Pre-compilation Clauses (PRECPL)	***ARCAD**, *ALDON
	Source Line Date (SRCDATE)	***CURRENT**, *ZERO, *KEEP
	Mark the conversion type (FLGCVTTYPE)	*YES, ***NO**, *KEEP
	Clean Temporary Cross-reference (CLRXREF)	**YES**, NO
	Clean Modified Lines (CLRFRMCHG)	**YES**, NO

Table 32: Summary of conversion options

Replace Existing Member (REPLACE)

If you defined a destination file and source member that already exists, choose whether the contents should be replaced.

Parameter	Description
YES	The contents of the destination source member are replaced by the converted source.
NO	The conversion is not done if the destination source member already exists.

Table 33: Replace Existing Member (REPLACE) parameters

Convert declaration specs (CVTDCLSPEC)

Define whether or not to convert the syntax of the (H, F, D and P) declaration specifications for the RPGLE and SQLRPGLE member.

> **Important!**
> This syntax option is only available from v7.2.0, via the installation of the Technology Refresh 7 PTF group (IBM i 7.2 TR 7).
>
> If you do not have this option at your RPGLE or SQLRPGLE compiler level, the conversion is not allowed.

Parameter	Description
YES	All declaration specifications in RPGLE (with some exceptions) are converted to Free syntax. In addition, C/Free and /End-Free clauses are all deleted. In the following cases, declaration specifications are not converted because there is no equivalent declaration in Free syntax: • In F specifications (pos. 18), use of a File Designation having the value P=Primary, S=Secondary, T=Array or Table, R=Address. • In F specifications (pos. 19), use of End of File. • In F specifications (pos. 21), use of Sequence. • In F specifications (pos. 28), use of Limits Processing. • In F specifications (pos. 34), value of Record Address Type not blank and not A.

Table 34: Convert declaration specs (CVTDCLSPEC) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
	• In F specifications (pos. 35), value of (Record Address File) for the File Organization. • In D specifications, use of keywords FROMFILE or TOFILE for the field names. For an F specification using an output file with add (O in pos. 17, A in pos. 20), the conversion includes the removal of the ADD keyword in the O specification. In addition, a declaration specification is not converted if a variable or procedure with a name that exceeds 99 characters is used. For a COPY clause the following rule is applied: • If it only contains DS subfields, no End-Ds is added at the end. • In all other cases, an End-Ds is added at the end. You must manually intervene for source lines that use the COPY clause, if the DS continues on lines that follow the /COPY.
NO	The (H, F, D and P) declaration specifications remain in classic syntax.

Table 34: Convert declaration specs (CVTDCLSPEC) parameters

Convert to Fully-Free (FULLYFREE)

Define whether or not to convert the source member to Fully-Free which ensures that the source lines can occupy all the columns, starting from the first column, up to the source file record length.

Important!

This syntax option is only available in v7.2.0, via the installation of the Technology Refresh 3 PTF group (IBM i v7.2TR3). It is also available in Technology Refresh 7.1TR11). Normally, the conversion to Fully-Free will not be done unless all the source lines can be converted to Free syntax. (see Max nbr of not free blocks (MAXNOTFREE))

If you do not have this option for your RPGLE or SQLRPGLE compiler level, the conversion is not allowed.

It is also necessary to request the conversion of the declaration specifications.

Parameter	Description
YES	The source is converted to Fully-Free. The special **FREE directive is added as the first line and the lines of code are shifted to the left (see First Column (Fully-Free) (FIRSTCOL)). If you want to keep the comments that are in columns 1-5, select ***KEEP** for the parameter Mark the conversion type (FLGCVTTYPE). They are then moved to the end of the line. The data lines for the compile time arrays and tables at the end of the compilation source, after the ** or **CTDATA directives are not affected by these modifications.
NO	The conversion to Fully-Free syntax is not done.

Table 35: Convert to Fully-Free (FULLYFREE) parameters

Max nbr of not free blocks (MAXNOTFREE)

If you selected ***YES** for the Convert to Fully-Free (FULLYFREE) parameter, choose whether or not to combine Fully-Free syntax with traditional syntax.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

In principle, a source that has been converted to Fully-Free syntax cannot contain any source line that is in traditional syntax. However, ARCAD-Transformer RPG allows the possibility to obtain a source that is almost Fully-Free, but has some blocks that remain in traditional syntax.

> ***Warning!***
> IBM does not officially support this! It is not recognized in the RDi 9.5 editor but it is supported by the RPGLE compiler.
>
> In order to get a real Fully-Free source, editable using RDi, you must modify these blocks of lines either by *rewriting them in Free syntax* (mainly for C-Specs) or by *moving them to /COPY clauses* (mainly for F-, I- or O-Specs).

Parameter	Description
***NONE**	No source lines in traditional syntax are permitted. If at least one source line cannot be converted to Free syntax, then the source is not converted to Fully-Free syntax.

1-999	Enter the maximum number of source line blocks not converted to Free syntax. They will be enclosed in special directives to allow compilation.
	Example
	`**END-FREE`
	`C GOTO TAG07`
	`**FREE`

*NOMAX	No limit is put on the maximum number of source line blocks that remain in traditional syntax.

Table 36: Max nbr of not Free blocks (MAXNOTFREE) parameters

First Column (Fully-Free) (FIRSTCOL)

If you selected ***YES** for the Convert to Fully-Free (FULLYFREE), choose on which column to start the source line.

> **Note**
> When the lines are indented, they are shifted to the right.

Parameter	Description
1	By default, the non-indented source lines start in column 1 (as opposed to column 8 in Free syntax).
2-5	The non-indented source lines start in column 2 to 5.

Table 37: First Column (Fully-Free) (FIRSTCOL) parameters

Convert Program calls (CVT_CALL)

Define whether or not to convert all traditional program calls (CALL) or procedure calls (CALLB), along with the entry/exit parameter declarations via *ENTRY PLIST to Free syntax.

For this the prototypes are created in the D-specs for each different program or procedure call (or if the parameter types/lengths are different); the name of the prototype starts **Pgm_** if it is for a program and **Prc_** if it is for a procedure.

After that, each traditional call is replaced by a prototyped call, with the parameters that were specified with PARM.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

For the parameters of the source itself (that were on the *ENTRY PLIST), a procedure interface (Pi) is also created; it takes the same name for the variables that were specified in PARM statements. However, there is a special case, when the variable in PARM is a DS: in this case another variable **ds_PiParm_nnn** is defined in the procedure interface, and the DS points to it via pointer pds_PiParm_nnn set at the start of the program.

It may be that these variables are already defined in D-specs as well: in which case, their previous definition is deleted (except if they are in a COPY clause, which will cause an error when compiling the new source).

If factors 1 and 2 were used for the PARM, then assignment instructions are added before/after the CALL or at the start and/or end of execution (for *ENTRY PLIST); if the execution is terminated with RETURN instructions assignment instructions will be inserted where necessary.

If PLIST / PARM statements are defined, they are deleted.

Parameter	Description
YES	CALL / CALLB instructions are replaced by prototyped calls. If *ENTRY PLIST is used, the program itself is prototyped. **Note** When an indicator LR (position 7576) was defined on the CALL/CALLB instruction, it was rarely voluntary: in that case, the following warning is inserted in the source, but the conversion is done without managing this indicator: // *CVTWNG. The *INxx indicator was defined on column 7576 for the CALL: Removed. Only the following cases prevent the replacement of CALL / CALLB with prototyped calls: • Indicator LR (position 75-76) is defined on the CALL / CALLB instruction. • A PLIST is used in a COPY clause. • An *ENTRY PLIST contains a field declared with SQLTYPE (BLOB, CLOB, DBCLOB).
NO	All CALL, CALLB, PLIST, PARM instructions remain in traditional syntax.

Table 38: Convert Program calls (CVT_CALL) parameters

Convert GoTo (CVT_GOTO)

Traditional branching opcodes GOTO / TAG / CABxx have no exact equivalent in Free syntax; normally it is necessary to restructure the code in order to transform it to structured programming.

However, the `ACVTRPGFRE` command allows the possibility to remove almost all these branching instructions in certain cases:

- When possible, using LEAVESR, LEAVE, ITER.
- By simulating the branching management in structured programming via the introduction of a variable containing the old label.

Of course, the structure of your program cannot be redesigned automatically, but this has the advantage of converting almost everything to Free syntax but the following cases always stay in traditional syntax, with GOTO/TAG:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- Branching from a subroutine to a label situated in the main body of the program (or procedure).
- Branching not respecting the nesting of structured programming (from outside of a loop or test/compare to within it).

Example

```
        If      Cdt1
                   Goto    Act02
                   Endif
                   ...
                   If      Cdt2
          Act02    Tag
                   ...
                   EndIf
```

- In SQLRPGLE, branching made in an SQL instruction WHENEVER ... GO TO label.
- Branching in the analyzed source to a label defined in a COPY clause.
- Branching from a COPY clause to a label defined in the analyzed source.
- Branching to a label that is not situated at the same control-level indicator level L0, L1..L9, LR (or later).
- Branching done inner a block of lines surrounded by conditional compilation directives (/IF ... /ENDIF).

Parameter	Description
*NO	All GOTO/TAG instructions remain in traditional syntax. ENDSR instructions with a label are split into 2 instructions (a TAG remaining in traditional syntax and an ENDSR transformed to Free syntax). CABxx instructions with a label are split into several instructions (test/comparison, management of any indicators, and a GOTO which remains in traditional syntax).
*BASE	Only branching operations that can be done directly in structured programming are modified: • Branching to a label situated on an ENDSR instruction: replaced by LEAVESR. • Branching to a label defined by a TAG situated just before the ENDSR instruction: replaced by LEAVESR. • Branching to a label situated just before an unconditioned end-of-loop (ENDDO/ENDFOR): replaced by ITER. • Branching to a label situated just after an end-of-loop (ENDDO/ENDFOR): replaced by LEAVE. • Unnecessary branching (GOTO) instructions to labels that are located immediately afterwards are deleted. • Unused labels (TAG) are deleted.
***ADVANCED**	In addition to the *BASE case, replacing most of the branching by structured programming tests, with the help of a variable containing the label name, and conditioning lines of code to get to the place where the label was defined. ***Example*** Branching to labels situated later in the code.

Table 39: Convert GoTo (CVT_GOTO) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description

	When the label is situated earlier in the code, an instruction is inserted allowing it to loop back (DoU).
	
	Before:
	
	`C If Cdt2`
	`C Goto TRT02`
	`C EndIf`
	`C If Cdt1`
	`C Goto Trt01`
	`C EndIf`
	`C Eval X = 0`
	`...`
	`C TRT01 Tag`
	`C Eval X = 1`
	`...`
	`C TRT02 Tag`
	`C Eval X = 2`
	`C ...`
	
	After:
	
	`If Cdt2;`
	`ATag = 'TRT02';`
	`EndIf;`
	`If ATag = *Blanks;`
	`If Cdt1;`
	`ATag = 'TRT01';`
	`EndIf;`
	`EndIf;`
	`If ATag = *Blanks;`
	`X = 0;`
	`...`
	`EndIf;`
	`If ATag = 'TRT01' or ATag = *blanks;`
	`ATag = *Blanks;`
	`X = 1;`
	`...`
	`EndIf;`
	`// branch when ATag = 'TRT02'`
	`ATag = *Blanks;`
	`X = 2;`
	
	Example
	Branching to labels situated previously in the code.
	
	For more complex cases (branching and labels situated at different DOxxx/FOR loop levels; existence of LEAVE/ITER instructions at the location where 'DoU' instructions are inserted to loop back in the code; etc.), this management sometimes requires:
	
	• the setting of the variable ATag with values *LEAVE or *ITER,
	• a request to exit from a loop-level with LEAVE,
	• a comparison/test of values in the loop variable after the end of a loop, to loop-back or exit again

Table 39: Convert GoTo (CVT_GOTO) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description

	to another loop-level.
	
	Before:
	
	`C  If  Cdt3`
	`C  Goto  TRT03`
	`C  EndIf`
	`C  TRT00  Tag`
	`C  If  Cdt1`
	`C  Goto  Trt01`
	`C  EndIf`
	`C  Eval  X = 0`
	`...`
	`C  TRT01  Tag`
	`C  Eval  X = 1`
	`...`
	`C  TRT02  Tag`
	`C  Eval  X = 2`
	`C  If  Cdt0`
	`C  Goto  TRT00 <===`
	`C  EndIf`
	`C  ...`
	`...`
	`C  TRT03  Tag`
	`C  Eval  X = 3`
	`C  If  Cdt2`
	`C  Goto  TRT02 <===`
	`C  EndIf`
	`C  ...`
	
	After:
	
	`If Cdt3;`
	`  ATag = 'TRT03';`
	`EndIf;`
	`DoU ATag <> 'TRT00' and ATag <> 'TRT01'`
	`  If ATag = 'TRT00' or ATag = *blanks;`
	`  ATag = *Blanks;`
	`  If Cdt1;`
	`  ATag = 'TRT01';`
	`  EndIf;`
	`  EndIf;`
	`  If ATag = *Blanks;`
	`  X = 0;`
	`  ...`
	`  EndIf;`
	`  If ATag = 'TRT01' or ATag = *blanks;`
	`  ATag = *Blanks;`
	`  X = 1;`
	`  ...`
	`  EndIf;`
	`  If ATag = 'TRT02' or ATag = *blanks;`
	`  ATag = *Blanks;`
	`  X = 2;`
	`  If Cdt0;`
	`  ATag = 'TRT00';`
	`  Iter;  <===`
	`  EndIf;`
	`  ...`

Table 39: Convert GoTo (CVT_GOTO) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description

	`EndIf;`
	`// branch when ATag = 'TRT03'`
	`ATag = *Blanks;`
	`X = 3;`
	`If Cdt2;`
	`  ATag = 'TRT02';`
	`  Iter;  <===`
	`EndIf;`
	`...`
	`EndDo;`

Table 39: Convert GoTo (CVT_GOTO) parameters

GOTO Label (TAGFLDNAME)

If you select ***ADVANCED** for the Convert GoTo (CVT_GOTO) parameter, define the name of a new variable that will be created (if necessary) in the source, to memorize and test for the old label name.

Enter a name with lower-case letters, ensuring it is syntactically correct.

If the name is already used in your program (example: **ATag**), then a name will be generated by adding 2 digits: ATag01 or ATag02.

Convert Key List (CVT_KLIST)

Define whether or not to convert the key lists (KLIST / KFLD) defined and used in the source to be converted.

There is no exact equivalent for KLIST / KFLD in Free syntax, but it is possible to indicate several key values directly in the file access instructions (CHAIN, SETLL, READE...).

Parameter	Description

***YES**	KLIST (+ KFLD) definitions are deleted. In all the lines where KLIST instructions were used, they are replaced by the list of variables that were defined with KFLD (even if these lines were already in Free syntax).
	Example
	Before:
	`C  KORDDET  KLIST`
	`C  KFLD  WOrdHdr`
	`C  KFLD  a_Line(I)`
	`...`
	`C  KORDDET  CHAIN  ORDDETF1`
	After:
	`Chain (WOrdHdr : a_Line(I)) ORDDETF1;`
	KLIST / KFLD specifications are only preserved if the KLIST / KFLD is used in at least one COPY clause.

*NO	The KLIST (+ KFLD) definitions are left in traditional syntax. They are still used in the file access instructions converted to Free syntax.

Table 40: Convert Key List (CVT_KLIST) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Convert MOVEA (CVT_MOVEA)

The classic operation code MOVEA, which assigns values with array variables, considered global strings which group all elements, has no exact equivalence in Free-Form syntax.

However, it is possible to manage their conversion:

- Assign a figurative constant.
- Assign between 2 arrays with same type/length.
- Use the %SubArr B.i.f.
- Redefine of the arrays as strings, using pointers.

Only the following cases cannot be converted (when they might be managed using a pointer):

- Arrays which total length of all the elements exceeds the maximum length allowed for a character string (65535 or 16383), when you are using v5.4.
- Arrays used as MOVEA result factors, but defined in the prototype of the program or procedure, with the CONST keyword.

Parameter	Description
*NO	All MOVEA instructions stay in traditional syntax.

*BASE	The following cases are done using a simple evaluation:
	• Assignment of a figurative constant (*BLANK, *ZERO, *LOVAL, *HIVAL, *ON, *OFF, *ALL' ') to all the elements of an array.
	• Assignment of a figurative constant (*ALL'xxxx', *ALLX'xxxxxx'), to all the elements of an array, when the length of the literal divides the length of one array element.
	• Assignment of a numeric value to all the elements of a numeric array.
	• Assignment between an alphanumeric value and an array element which length is upper or equal, without padding blanks for the rest of the array.
	
	The following cases are done using a partial evaluation of array(s), with %SubArr(...):
	• Assignment of a figurative constant (*BLANK, *ZERO, *LOVAL, *HIVAL, *ON, *OFF, *ALL' ') beginning from one array element.
	• Assignment of a figurative constant (*ALL'xxxx', *ALLX'xxxxxx'), beginning from one array element, when the length of the literal divides the length of one array element.
	• Assignment between 2 arrays which have the same element lengths (for all the elements or just some elements).
	
	Example
	MOVEA is used to assign a list of 0 or 1 values in the array of indicators, from one of its elements and for a maximum of 8 elements:
	`MOVEA 0100 *IN(15)`
	then it is replaced by several instructions
	`*INxx = '0';`

Table 41: Convert MOVEA (CVT_MOVEA) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
	or `*INxx = '1';`

*ADVANCED	In addition to the *BASE case, this option replaces most of the MOVEA with an instruction block using redefinition as string(s) via pointer(s), in order to run the operation with a simple %SUBST.
	
	To do this, work variables are defined (when needed):
	• AFrmArrStr & pAFrmArrStr: redefinition as a string and pointer for an array used in Factor 2.
	• NFrmArrStrLen: Variable for the number of characters to be taken into account in the array used in Factor 2.
	• AToArrStr & pAToArrStr: redefinition as a string and pointer for an array used in result factor.
	• NToArrStrLen: Variable for the number of characters to be taken into account in the array used in result factor.
	• NChgArrStrLen: Variable for the number of characters to change in the array used in result factor.
	• If the variables are Unicode type, they are CFrmArrStr, etc.
	• If the variables are Graphic type, they are GFrmArrStr, etc.
	
	The corresponding instruction block in Free-Form may have any number of instructions:
	
	Example
	Assignment of array elements to a string which is longer
	
	Before:
	`D a_A01 s 1a Dim(100)`
	`D B02 s 2a`
	`...`
	`C Movea a_A01(4) B02`
	
	After:
	`pAFrmArrStr = %Addr(a_A01(4));`
	`%Subst(B02:1:2) = %Subst(AFrmArrStr:1:2);`
	
	Example
	Assignment between 2 arrays that have different element lengths, with variables as indexes
	
	(this case needs the most instructions)
	
	Before:
	`D a_A01 s 1a Dim(100)`
	`D a_B02 s 2a Dim(100)`
	`...`
	`C Movea a_A01(X1) a_B02(X2)`

Table 41: Convert MOVEA (CVT_MOVEA) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description

	After:
	`NFrmArrStrLen=(%Elem(a_A01)+1-X1)*1;`
	`NToArrStrLen=(%Elem(a_B02)+1-X2)*2;`
	`NChgArrStrLen = NFrmArrStrLen;`
	`If NChgArrStrLen > NToArrStrLen;`
	`NChgArrStrLen = NToArrStrLen;`
	`EndIf;`
	`pAFrmArrStr = %Addr(a_A01(X1));`
	`pAToArrStr = %Addr(a_B02(X2));`
	`%Subst(AToArrStr:1:NChgArrStrLen)`
	`= %Subst(AFrmArrStr:1:NChgArrStrLen);`

Table 41: Convert MOVEA (CVT_MOVEA) parameters

Convert Subr. to procedures (CVT_SUBR)

Define whether or not to convert the subroutines in the main part of the program to ILE procedures.

In the old source, subroutines (and their calls) are converted whether they use the old columned syntax or the new Free syntax.

Example

Before:

```
C              Exsr      SrAmount
C              ....
C   SrAmount   BegSr
C
C              EndSr
```

After:

```
SrAmount();
...
Dcl-Proc SrAmount;
...
End-Proc SrAmount;
```

However, the new ILE procedures have no parameters and no local variables; they continue to use global variables in your source.

You can only request this change if you also convert the calculation specifications (Convert calculation specs (CVTCLCSPEC)[***FREE**]) - and the declaration specifications (Convert declaration specs (CVTDCLSPEC)[***YES**]) to Free-Form.

When you do not convert declaration specifications to Free-Form CVTDCLSPEC(*NO), for each new procedure, it will create DSpecs for the prototype and procedure interfaces and PSpecs for the beginning and end of each new procedure. This will happen when you compile programs for v5.4 or v6.1 OS level.

Warning!

When the created object type is a *PGM, in order to use the ILE procedures, your program must be genuinely ILE before being executed. Specifically, it must no longer use the default activation group. If this is not the case, it will be necessary to recompile this source with DFTACTGRP(*NO), defining a value for the ACTGRP parameter.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

This may change the execution context of your program and you may need to modify the scope of any OVRDBF commands executed in the calling programs.

Parameter	Description
*YES	Most subroutines are converted into procedures. However, the following subroutines cannot be converted to ILE procedures and therefore remain as subroutines: • Subroutines already located in an ILE procedure outside the main body of the source; • Subroutines using a GOTO with a label (TAG) located outside the subroutine; • Subroutines using a RETURN operation code, in order to exit from the program; • Special subroutines *INZSR and *PSSR; • Subroutines that are defined as file error management subroutines (keyword INFSR). Subroutines using a file with the INFSR keyword (otherwise error message RNF5416 will be issued on compilation). All subroutines that call a subroutine that cannot itself be converted (recursively). If necessary, the new procedures are moved: • after the routines that cannot be processed. • after O specifications. **Note** In SQLRPGLE, no subroutine is processed if the SQL statement WHENEVER ... GO TO label is used.
***NO**	Subroutines are not converted into procedures.

Table 42: Convert Subr. to procedures (CVT_SUBR) parameters

Use %ParmNum (USEPARMNUM)

In v7.1, a new BiF. %ParmNum(Param_Name) has been introduced, in order to avoid having to hard-code the parameter number - compare to %Parms. Additionally, when new parameters are added later, this avoids possible modification errors.

Parameter	Description
***YES**	When comparing the BiF. %Parms and an integer is found, the integer is replaced by the equivalent BiF. %ParmNum(Param_Name). • If the comparison is type < n, it is treated as <= n-1 • If the comparison is type > n, it is treated as >= n+1 This modification is only made if you have also requested the conversion of calculation specifications. It is then applied to comparison instructions already coded in Free syntax. This also works when the number of received parameters is retrieved using a variable defined in the SDS in positions 37 to 39 (or with *PARMS). **Note** This new syntax option is only available in v7.1.0. If you do not have this option at your RPGLE or SQLRPGLE compiler level, the modification is not performed.

Table 43: Use %ParmNum (USEPARMNUM) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
*NO	The comparisons using %Parms are not modified.

Table 43: Use %ParmNum (USEPARMNUM) parameters

Indentation Size (char) (INDENT)

Define the indentation to be applied to the Free source, according to the programming operation structures used (IF, SELECT, ... ENDxx).

> **Note**
> This does not affect the instructions already in Free syntax.
>
> The indentation does not change for the instructions that are included after the compilation directives /ELSE, /ELSEIF or in 2 levels of /IF.

Parameter	Description
0	No indentation is performed.
1-5	Number of blanks added for each new indentation.

Table 44: Indentation Size (char) (INDENT) parameters

Indent Comments (INDENTCMT)

Define whether or not to indent the comments preceding indented instructions.

> **Note**
> This does not affect the comments already in Free syntax.

Parameter	Description
***YES**	If there is enough space, comments are indented and aligned with the following instructions.
*NO	Converted comments are not indented - they begin in column 8.

Table 45: Indent Comments (INDENTCMT) parameters

Empty Comment Lines (EMPTYCMT)

Define the action to be performed when blank comment lines are encountered, whether it is:

- an * in column 7,
- a //,
- the specification letter in column 6, or
- blank lines.

> **Note**
> If there are characters in columns 1-5 for these rows, they are ignored for the tests.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
***KEEP**	All lines are preserved; the empty comment lines (with an * in column 7) are converted to Free comments (//).
*BLANK	All lines are preserved; the empty comment lines (with an * in column 7) or the comments already in Free-Form (//) are entirely set to blank.
*ONELINE	The empty comment lines (with an * in column 7) or the comments already in Free-Form (//) are entirely set to blank. If multiple blank lines follow each other, then only one is retained.
*REMOVE	All the following lines are deleted: • lines with just one * in column 7; • lines with an empty comment in Free syntax (//), and • lines that are already blank.

Table 46: Empty comment lines (EMPTYCMT) parameters

Keep indentation in the DS (KEEPDSIND)

Define whether or not to keep the existing indentation for different fields of a Data Structure, as they are specified in the D specifications.

Parameter	Description
*YES	The names of the fields that are part of a DS keep their indentation, as in the original source.
***NO**	The names of all the fields that make up a DS are aligned. **Note** If the field names started in column 7, they are aligned with those starting in column 8.

Table 47: Keep indentation in the DS (KEEPDSIND) parameters

Case for operation codes (OPCODECASE)

Choose the case for the calculation operation codes converted to Free syntax (e.g. EvalR, Chain, When...).

Note
This does not affect the instructions already in Free syntax.

Parameter	Description
***MIXED**	Operation codes are written in mixed case: the first character of each word of the operation code is in upper case and the other characters are in lower case.
*UPPER	All operation codes are written in upper case.
*LOWER	All operation codes are written in lower case.

Table 48: Case for operation codes (OPCODECASE) parameters

Case for the B.i.F. (BLTFNCCASE)

Choose the case for the built-in-function names on lines converted to Free syntax (e.g. %Subst, %Len,

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

%Scan...).

> **Note**
> This does not affect the instructions already in Free syntax.

Parameter	Description
***MIXED**	Built-in-function names are written in mixed case: the first character of each word in the built-in-function name is in upper case and the other characters are in lower case.
*UPPER	Built-in-function names are written in upper case.
*LOWER	Built-in-function names are written in lower case.

Table 49: Case for the B.i.F. (BLTFNCCASE) parameters

Case for special words (SPCWRDCASE)

Choose the case for the special words used in calculation specifications converted to Free syntax (e.g. *Blank, *Null, *Zero...).

> **Note**
> This does not affect the instructions already in Free syntax.

Parameter	Description
***MIXED**	Special words are written in mixed case: the first character of each word in the operation code is in upper case and the other characters are in lower case.
*UPPER	Special words are written in upper case.
*LOWER	Special words are written in lower case.

Table 50: Case for special words (SPCWRDCASE) parameters

Case for key words (KEYWRDCASE)

Choose the case for the keywords used in declaration specifications converted to Free syntax (e.g. DclDs, Char, Keyed...)

> **Note**
> This does not affect the instructions already in Free syntax nor the keywords already in H, F, P or D specifications.

Parameter	Description
***MIXED**	Keywords are written in mixed case: the first character of each word in the operation code is in upper case and the other characters are in lower case.
*UPPER	All keywords are written in upper case.
*LOWER	All keywords are written in lower case.

Table 51: Case for key words (KEYWRDCASE) parameters

CHECK (CHECKIND)

Define whether or not to analyze the risks due to the fact that the %Check (or %CheckR) BIF, used to convert the CHECK (or CHECKR) operation code, does not update the %Found or %Equal indicators.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

When the specified value is ***WNG1** or *YES, the analysis for using the %Found or %Equal indicators in the same source is performed. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

- to replace the use of %Found or %Equal indicators with a test based on the value of the position or the result index (this test is only done automatically if it appears on the instruction that immediately follows the converted CHECK operation code), or
- to insert a warning comment right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***WNG1** or *YES values specified in this parameter.

If no action is entered for a source line, no replacement of %Found or %Equal is performed on the next instruction and, depending on the value specified in this parameter, a comment is added (or not) in the converted source.

Parameter	Description
***WNG1**	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. In addition, a comment is added to the converted source.
*YES	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible use in the source of %Found or %Equal indicators is not analyzed. No warning message is issued.

Table 52: Analyze Indicator Problems CHECK (CHECKIND) parameters

SCAN (SCANIND)

Define whether or not to analyze the risks due to the fact that the %Scan BIF, used to convert the SCAN operation code, does not update the %Found or %Equal indicators.

When the value is ***WNG1** or *YES, the analysis for using the %Found or %Equal indicators in the same source is performed. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

- to replace the use of %Found or %Equal indicators with a test based on the value of the position or the result index (this test is only done automatically if it appears on the instruction that immediately follows the converted SCAN operation code), or
- to insert a warning comment right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***WNG1** or *YES values specified in this parameter.

If no action is entered for a source line, no replacement of %Found or %Equal is performed on the next instruction and, depending on the value specified in this parameter, a comment is added (or not) in the converted source.

Parameter	Description
***WNG1**	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. In addition, a comment is added to

Table 53: Analyze Indicator Problems SCAN (SCANIND) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
	the converted source.
*YES	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible use in the source of %Found or %Equal indicators is not analyzed. No warning message is issued.

Table 53: Analyze Indicator Problems SCAN (SCANIND) parameters

LOOKUP (LOOKUPIND)

Define whether or not to analyze the risks due to the fact that the %LookUpXX (or %TLookUpXX) BIF, used to convert the LOOKUP operation code, does not update the %Found or %Equal indicators.

When the specified value is ***WNG1** or *YES, the analysis for using the %Found or %Equal indicators in the same source is performed. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

- to replace the use of %Found or %Equal indicators with a test based on the value of the position or the result index (this test is only done automatically if it appears on the instruction that immediately follows the converted LOOKUP operation code), or
- to insert a warning comment right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***WNG1** or *YES values specified in this parameter.

If no action is entered for a source line, no replacement of %Found or %Equal is performed on the next instruction and, depending on the value specified in this parameter, a comment is added (or not) in the converted source.

Parameter	Description
***WNG1**	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. In addition, a comment is added to the converted source.
*YES	The possible use in the source of %Found or %Equal indicators is analyzed. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible use in the source of indicators %Found or %Equal is not analyzed. No warning message is issued.

Table 54: Analyze Indicator Problems LOOKUP (LOOKUPIND) parameters

Z-ADD, Z-SUB (NUMTRUNCZ)

Define whether or not to analyze the risks of numeric truncation due to the conversion of the following assignment operation codes or not: Z-ADD, Z-SUB, and PARM (when factor 1 or factor 2 are specified).

For these assignment operation codes, it is possible that the developer intentionally performed a numeric truncation in order to extract the last digits.

When the specified value is ***YES**, *WNG1 or *WNG2, the analysis for risks of truncation is performed for numeric fields. The warnings issued can be accessed and managed in the plug-in.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

For each instruction, you can choose:

- to replace the official converted instruction with a more complex one, in order to take into account the risk of numeric truncation, or
- to insert a warning comment (one or two lines) right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***YES**, *WNG1 or *WNG2 values specified in this parameter.

If no action is entered for a source line, then the official converted instruction remains simple and the risk of truncation is considered unlikely. Depending on the value specified in this parameter, a comment of one or two lines is added (or not) in the converted source.

Parameter	Description

*WNG1	When a risk of numeric field truncation is detected, a warning message is added as a comment in the converted source, right after the converted instruction.
	If the operation code has no factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Ope-Code Type(len,dec)->Type(len,dec)`
	If the operation code has a factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Type(len,dec) Ope-Code Type(len,dec)->Type(len,dec)`

*WNG2	When a risk of numeric field truncation is detected, 2 comment lines are added in the converted source, right after the converted instruction. • The first line displays the same comment as for *WNG1. • The second line displays an alternative instruction to get the same result while avoiding numeric truncation. If necessary, you can manually replace the current instruction with the alternative instruction displayed in the second comment line.
***YES**	The possible risk of numeric field truncation is detected. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible risk of numeric field truncation is not detected. No warning message is issued.

Table 55: Analyze Numeric Truncation Z-ADD, Z-SUB (NUMTRUNCZ) parameters

ADD, SUB {Length(Fact1/Fact2)>Length(Result)} (NUMTRUNCA)

Define whether or not to analyze the risks of numeric truncation due to the conversion of the following addition and subtraction operation codes, when the length of factor 1 or factor 2 is greater than the length of the result: ADD and SUB.

For these addition and subtraction operation codes, it is unlikely that the developer intentionally performed a numeric truncation in order to extract the last digits.

When the specified value is ***YES**, *WNG1 or *WNG2, the analysis for risks of truncation is performed for numeric fields. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- to replace the official converted instruction with a more complex one, in order to take into account the risk of numeric truncation, or
- to insert a warning comment (one or two lines) right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***YES**, *WNG1 or *WNG2 values specified in this parameter.

If no action is entered for a source line, then the official converted instruction remains simple and the risk of truncation is considered unlikely. Depending on the value specified in this parameter, a comment of one or two lines is added (or not) in the converted source.

Parameter	Description

*WNG1	When a risk of numeric field truncation is detected, a warning message is added as a comment in the converted source, right after the converted instruction.
	If the operation code has no factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Ope-Code Type(len,dec)->Type(len,dec)`
	If the operation code has a factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Type(len,dec) Ope-Code Type(len,dec)->Type(len,dec)`

*WNG2	When a risk of numeric field truncation is detected, 2 comment lines are added in the converted source, right after the converted instruction. • The first line displays the same comment as for *WNG1. • The second line displays an alternative instruction to get the same result while avoiding numeric truncation. If necessary, you can manually replace the current instruction with the alternative instruction displayed in the second comment line.
***YES**	The possible risk of numeric field truncation is detected. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible risk of numeric field truncation is not detected. No warning message is issued.

Table 56: Analyze Numeric Truncation ADD, SUB {Length(Fact1/Fact2)>Length(Result)} (NUMTRUNCA) parameters

ADD, SUB {Other} (NUMTRUNCB)

Define whether or not analyze the risks of numeric truncation due to the conversion of the following addition and subtraction operation codes, when the length of factor 1 and factor 2 is not greater than the length of the result, but it is either that the factor 1 is not specified or that the length of factor 1 or factor 2 is equal to the length of the result. A risk of exceeding the field size (MCH1210) exists if there is a remainder for the following operation codes: ADD and SUB.

For these addition and subtraction operation codes, the default value is ***NO** since it is extremely unlikely that the developer intentionally performed a numeric truncation in order to extract the last digits.

When the specified value is *YES, *WNG1 or *WNG2, the analysis for risks of truncation is performed for numeric fields. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- to replace the official converted instruction with a more complex one, in order to take into account the risk of numeric truncation, or
- to insert a warning comment (one or two lines) right after the converted instruction.

If actions are entered for some source lines, they are substituted for the *WNG1, *WNG2 or *YES values specified in this parameter.

If no action is entered for a source line, then the official converted instruction remains simple and the risk of truncation is considered unlikely. Depending on the value specified in this parameter, a comment of one or two lines is added (or not) in the converted source.

Parameter	Description

*WNG1	When a risk of numeric field truncation is detected, a warning message is added as a comment in the converted source, right after the converted instruction.
	If the operation code has no factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Ope-Code Type(len,dec)->Type(len,dec)`
	If the operation code has a factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Type(len,dec) Ope-Code Type(len,dec)->Type (len,dec)`

*WNG2	When a risk of numeric field truncation is detected, 2 comment lines are added in the converted source, right after the converted instruction. • The first line displays the same comment as for *WNG1. • The second line displays an alternative instruction to get the same result while avoiding numeric truncation. If necessary, you can manually replace the current instruction with the alternative instruction displayed in the second comment line.
*YES	The possible risk of numeric field truncation is detected. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
***NO**	The possible risk of numeric field truncation is not detected. No warning message is issued.

Table 57: Analyze Numeric Truncation ADD, SUB {Other} (NUMTRUNCB) parameters

MULT (NUMTRUNCM)

Define whether or not to analyze the risks of numeric truncation due to the conversion of the multiplication (MULT) operation code, when the length of the integer result of the operation could be greater than the length of the result.

For this multiplication operation code, it is unlikely that the developer intentionally performed a numeric truncation in order to extract the last digits. However, it is possible that the operation code was used for date fields in order to shift digits by multiplying by 100, 10000, 0.01, and so on.

When the specified value is ***YES**, *WNG1 or *WNG2, the analysis for risks of truncation is performed for numeric fields. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- to replace the official converted instruction with a more complex one, in order to take into account the risk of numeric truncation, or
- to insert a warning comment (one or two lines) right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***YES**, *WNG1 or *WNG2 values specified in this parameter.

If no action is entered for a source line, then the official converted instruction remains simple and the risk of truncation is considered unlikely. Depending on the value specified in this parameter, a comment of one or two lines is added (or not) in the converted source.

Parameter	Description

*WNG1	When a risk of numeric field truncation is detected, a warning message is added as a comment in the converted source, right after the converted instruction.
	If the operation code has no factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Ope-Code Type(len,dec)->Type(len,dec)`
	If the operation code has a factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Type(len,dec) Ope-Code Type(len,dec)->Type (len,dec)`

*WNG2	When a risk of numeric field truncation is detected, 2 comment lines are added in the converted source, right after the converted instruction. • The first line displays the same comment as for *WNG1. • The second line displays an alternative instruction to get the same result while avoiding numeric truncation. If necessary, you can manually replace the current instruction with the alternative instruction displayed in the second comment line.
***YES**	The possible risk of numeric field truncation is detected. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible risk of numeric field truncation is not detected. No warning message is issued.

Table 58: Analyze Numeric Truncation MULT (NUMTRUNCM) parameters

DIV (NUMTRUNCD)

Define whether or not to analyze the risks of numeric truncation due to the conversion of the division (DIV) operation code, when the length of the integer result of the operation could be greater than the length of the result.

For this division operation code, it is unlikely that the developer intentionally performed a numeric truncation in order to extract the last digits. However, it is possible that the operation code was used for date fields in order to shift digits by dividing by 100, 10000, 0.01, and so on.

When the specified value is ***YES**, *WNG1 or *WNG2, the analysis for risks of truncation is performed for numeric fields. The warnings issued can be accessed and managed in the plug-in.

For each instruction, you can choose:

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

- to replace the official converted instruction with a more complex one, in order to take into account the risk of numeric truncation, or
- to insert a warning comment (one or two lines) right after the converted instruction.

If actions are entered for some source lines, they are substituted for the ***YES**, *WNG1 or *WNG2 values specified in this parameter.

If no action is entered for a source line, then the official converted instruction remains simple and the risk of truncation is considered unlikely. Depending on the value specified in this parameter, a comment of one or two lines is added (or not) in the converted source.

Parameter	Description

*WNG1	When a risk of numeric field truncation is detected, a warning message is added as a comment in the converted source, right after the converted instruction.
	If the operation code has no factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Ope-Code Type(len,dec)->Type(len,dec)`
	If the operation code has a factor 1, the comment is in the following format:
	`// *CVTWNG *NUMTRUNCx : Type(len,dec) Ope-Code Type(len,dec)->Type (len,dec)`

*WNG2	When a risk of numeric field truncation is detected, 2 comment lines are added in the converted source, right after the converted instruction. • The first line displays the same comment as for *WNG1. • The second line displays an alternative instruction to get the same result while avoiding numeric truncation. If necessary, you can manually replace the current instruction with the alternative instruction displayed in the second comment line.
***YES**	The possible risk of numeric field truncation is detected. The warnings issued can be accessed and managed in the plug-in. By default, no comment is added to the converted source.
*NO	The possible risk of numeric field truncation is not detected. No warning message is issued.

Table 59: Analyze Numeric Truncation DIV (NUMTRUNCD) parameters

Analyze Alpha to num. MOVE (ALPHTONUM)

Examine the inherent risks in assigning alphanumeric values to numeric fields, using the MOVE, MOVEL operation codes.

Any warnings are put into the *AARFCHWF1* file, which you can manage via ARCAD-Transformer RPG. You can choose to insert the warning in comments, just after the converted instruction, for every instruction.

> **Note**
> If actions are entered for certain source lines, they override the *YES or *WNG1 values specified for this option.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
*WNG1	For MOVE, MOVEL, each case where an alphanumeric value is assigned to a numeric field is examined. Any warnings issued can be managed and a comment is added to the converted source.
*YES	For MOVE, MOVEL, each case where an alphanumeric value is assigned to a numeric field is examined. Any warnings issued can be managed but no comment is added to the converted source.
***NO**	Cases where an alphanumeric value is assigned to a numeric field are not examined. No warnings are issued.

Table 60: Analyze Alpha to num. MOVE (ALPHTONUM) parameters

Pre-compilation Clauses (PRECPL)

Specifies which syntax has been used for the pre-compilation clauses added at the start of the source.

Parameter	Description
***ARCAD**	By default, the syntax of ARCAD pre-compilation clauses is used.
*ALDON	Aldon pre-compilation clause syntax is used.

Table 61: Pre-compilation Clauses (PRECPL) parameters

Source Line Date (SRCDATE)

Define how the date field is modified (for each source line) in the converted source.

Parameter	Description
***CURRENT**	For lines added or modified, the date for each record is set to the current date. For other non-modified lines, the previous date is retained.
*ZERO	The date is set to **000000** for all records in the new source.
*KEEP	For changed lines, the old date is kept. For added lines, the date is set to the current date. For other non-modified lines, the previous date is retained. **Important!** Some instructions that were using only one line in the old source may use several lines after conversion. In this case, only the first line will keep the old source date. The new lines will be set to the current date.

Table 62: Source Line Date (SRCDATE) parameters

Mark the conversion type (FLGCVTTYPE)

Choose whether or not to place an identifying mark for each type of conversion in columns 1-5 of the converted source.

Parameter	Description
*YES	In columns 1-5, a mark is added identifying the conversions as follows: • Gnnn: conversion of branching instructions (GOTO/TAG)

Table 63: Mark the conversion type (FLGCVTTYPE) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Description
	• Cnnn: conversion of program calls (CALL/CALLP) • Xnnn: conversion of other operation codes • Snnn: instruction not converted - no equivalent in Free syntax **Note** This is mainly for the development of conversions, or in case of conversion problems.
***NO**	Columns 1-5 of the new source are set to blank (for the added or modified lines).
*KEEP	The values (comments) in the old source in columns 1-5 are retained for added or modified lines (in most cases). However, any special color attribute characters are replaced by blanks. **Note** Keeping these comments in columns 1-5 can be annoying in RDi because they can appear in the active columns of the source, if we make a shift to the right using the RRn LPEX source editor command.

Table 63: Mark the conversion type (FLGCVTTYPE) parameters

Clean Temporary Cross-reference (CLRXREF)

Choose whether or not to delete X-Ref data generated at the start of the process and used for conversion of the RPGLE.

Parameter	Description
YES	The temporary X-Ref data is not conserved.
NO	The temporary X-Ref data is conserved at the end of processing. They are therefore never cleared.

Table 64: Clean Temporary Cross-references (CLRXREF) parameters

Clean Modified Lines (CLRFRMCHG)

Choose whether or not to delete the converted source lines added to ARCAD file *AARFCHSF1*.

Parameter	Description
YES	The lines of converted source put in file *AARFCHSF1* are not retained. This is the recommended value if you have specified a destination source member.
NO	The lines of converted source are retained at the end of the process in file *AARFCHSF1*. These lines will never be purged. This is the required value if you specified ***NONE** for the destination source member.

Table 65: Clean Modified Lines (CLRFRMCHG) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

10 Managing the default RPG to RPGLE conversion options

If your source code is in RPG form (not RPGLE), you must first convert it to RPGLE.

RPG to RPGLE conversion parameters are managed from the plug-in's preferences.

Parameter	Description
Temporary Library	The name of the library that contains the source file where the RPGLE converted source member will be generated (a location to store temporary objects, for example).
Source File	The source file where the RPGLE converted source member will be generated. This is the version that will later be converted to free.
Extended Parameters	Additional values that will be added as-is to the `CVTRPGSRC` command before its execution.
Create the Target Source File	If this option is checked, the source file defined in the **Source File** field will be automatically created if it does not exist.
Replace the Target Source Member	If this option is checked, any existing target source member with the same name as the current member created during the RPG to RPGLE conversion will be deleted (replaced) during the conversion. This is most likely the case if the conversion has already been performed in the past.

Table 66: The RPG to RPGLE conversion options

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Conversion process

11 Preparing the environment

Chapter summary

This section provides instructions for creating and connecting to a new IBM i connection, creating a new development library to contain modernized source code and creating a new source member.

Important!

You are not required to create a new profile or a new target library to use ARCAD-Transformer RPG, however you are required to use libraries and library lists to successfully transform data.

If you do create a new library, it is your responsibility to ensure the required objects, outlined in the following example(s), do not already exist on your system. If they are already present you will need to substitute values.

Note

The library called **ARCTW_SMPL** (the ARCAD Sample library) is used in the following example(s) and provided only if you use an ARCAD Server. If you are not using ARCAD, the stand-alone plug-in does not include this library.

11.1 Creating a connection to an IBM i server

A connection in RDi is roughly equivalent to a green screen session in IBM i.

Follow the subsequent steps to create a connection to your IBM i after installing and configuring ARCAD-Transformer RPG.

Step 1 Open the **Remote Systems** view (*Window* > *Show View* > *Remote Systems* > *Remote Systems*).

Step 2 Open the **New Connection** node.

Step 3 Right-click on **IBM i**, then select **New Connection**.

Step 4 Enter the required connection information in the **New Connection** dialog.

In the example below, the IBM i **Host name** is *10.100.10.190*, yours will be different. The **Description** field is optional.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

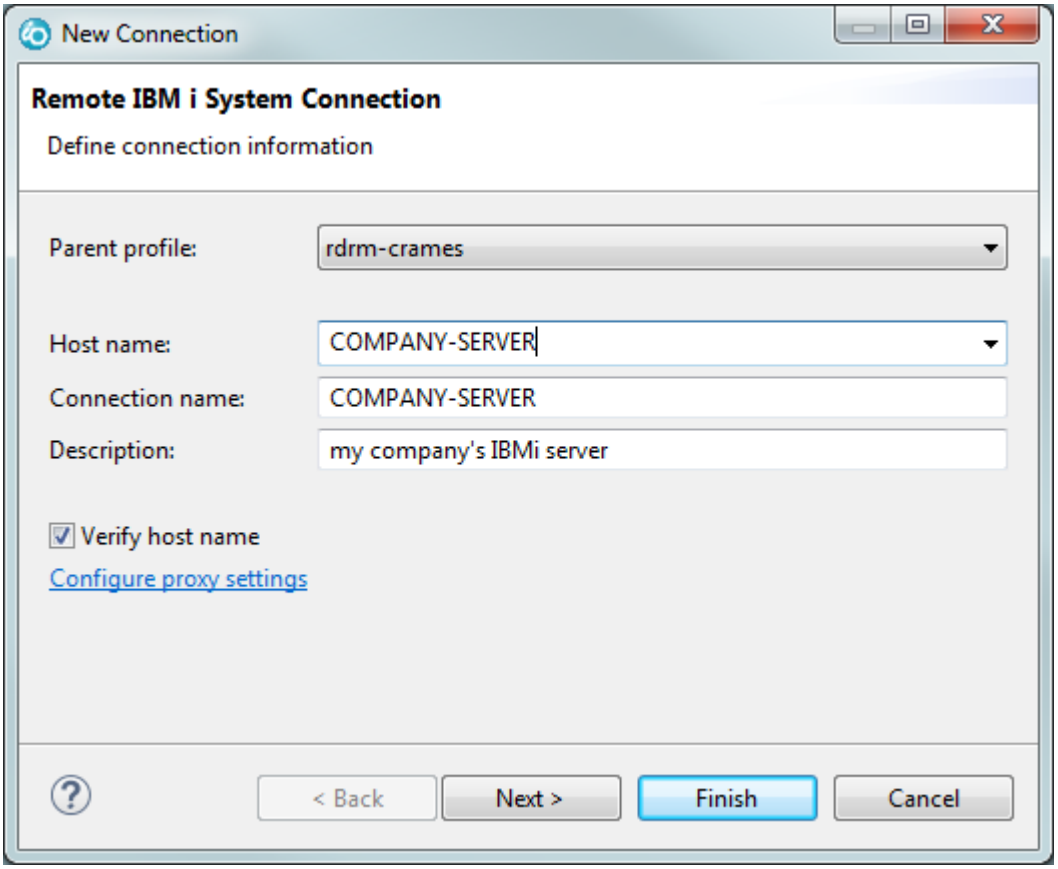


Figure 3: Example of a new IBM i connection

Step 5 Click **Finish**.

11.2 Logging in to a connection

When the connection created appears in the list of remote connections (**Remote Systems** view), you can connect to it.

Step 1 Right-click on the connection and select **Connect**.

Step 2 Enter your user ID and password in the login screen.

> **Note**
> You can have multiple active connections to the same physical system.

11.3 Creating source member filters

Filtering the source members to convert in your library(ies) can be very useful. You can regroup member types, for example, to easily find the files needed for modernization.

Follow the subsequent steps to create a source member filter.

Step 1 From the **Remote Systems** view, expand the connection for which you want to create a

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

new filter.

Step 2 Expand the **Objects** menu, then the **Work with members...** menu.

Result The **New Member Filter** dialog is displayed.

Step 3 Enter the following values to define the IBM i member to filter:

Library	This is the library the filter will apply to.
File	This is the source file in the defined library to filter.
Member filter	Leave the asterisk as is to search through all members.

The **Member text** and **Member type** fields should also be unfiltered to open the filter to a maximum of members available.

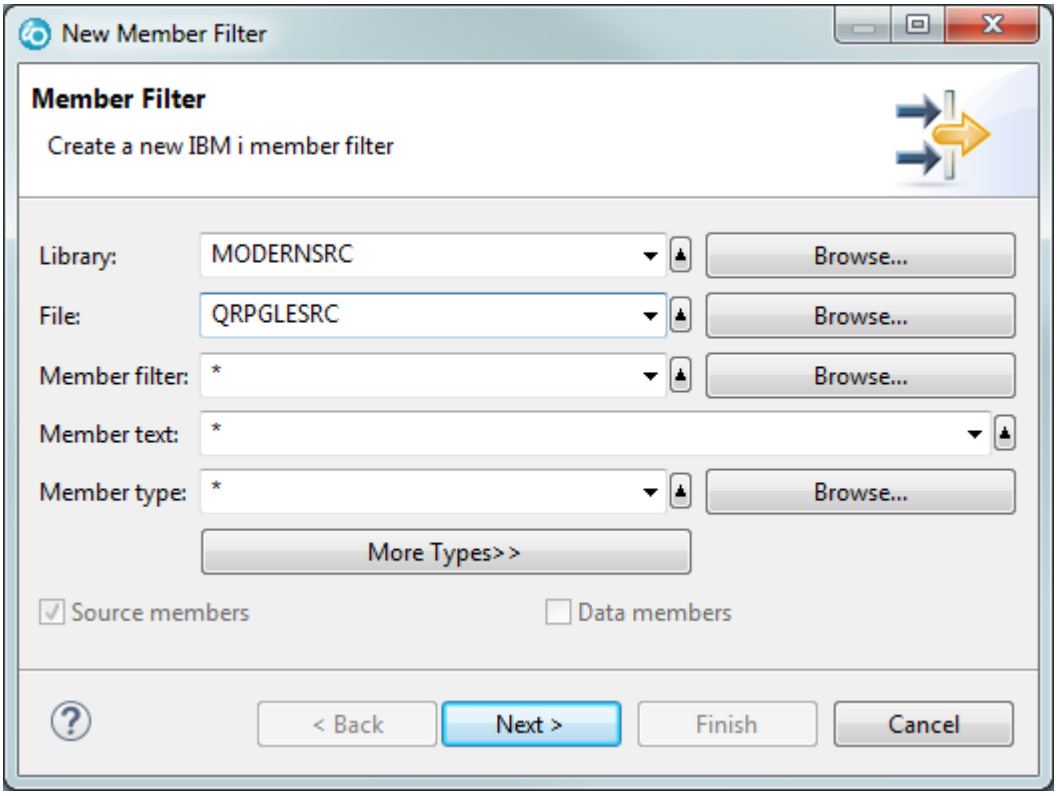


Figure 4: Member filter dialog page 1

Click **Next >** to continue.

Step 4 Enter the name of the new filter in the **Filter Name** field and ensure the **Only create filter in this connection** option is checked.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

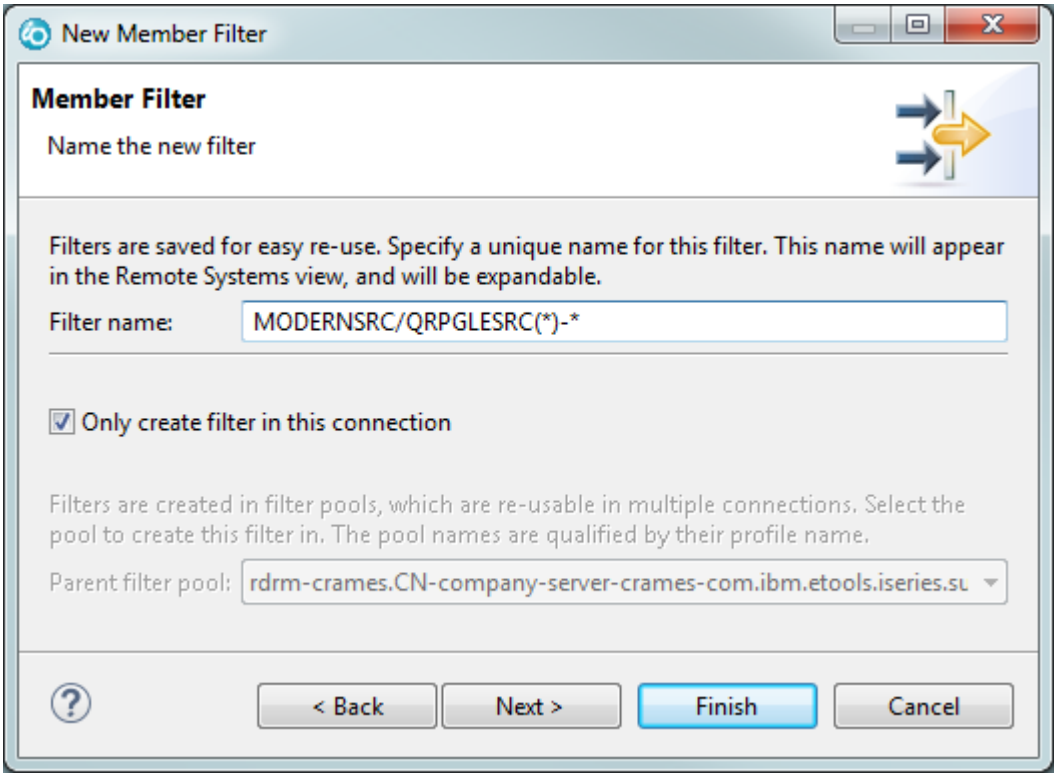


Figure 5: Member filter dialog page 2

Step 5 Click **Finish**.

Result The new filter is created and appears in the root of the **Objects** menu (*Objects* > *Filter*). Expanding the new filter displays all the files included in the scope of the filter.

11.4 Adding libraries to library lists

A well-structured library list is essential to successful source code conversion. Invalid library lists are often the cause of errors during conversion.

> **Note**
> A successful RDi source verification will eliminate many of your failed conversion attempts.

The conversion requires the same runtime environment as any program compile would.

For a source member to compile, your master RDi library list must contain all of the necessary libraries. This includes libraries that contain database files, binding directories, service programs, /COPY members etc. In the following example(s), only one library is required: the library we created to modernize - **MODERNSRC**.

There are two recommended ways to add libraries to a list for ARCAD-Transformer RPG. You can use the `ADDLIBLE` command in the command line, or use the **Objects** menu in the **Remote Systems** view.

11.4.1 Adding a library to a library list from the command line

Step 1 Open the

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Commands Log view (*Window* > *Show View* > *IBM i* > *Commands Log*).

Step 2 Ensure the command will be executed in the **Normal** mode.

Step 3 Enter `ADDLIBLE` in the **Command** field and click **Prompt...**

Result The **Add Library List Entry (ADDLIBLE)** dialog is displayed.

Step 4 Enter the required details for the list, then click **OK**.

Enter the name of the library to add to the list in the **Library** field.

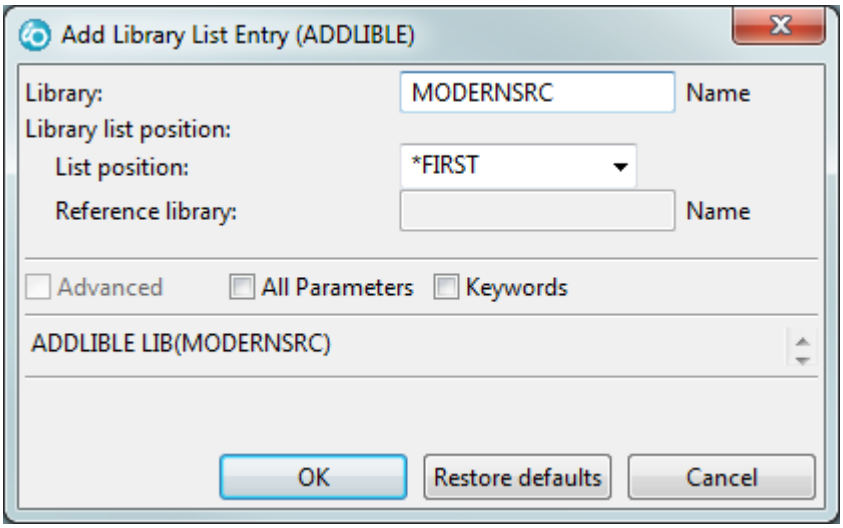


Figure 6: Create Library List dialog

Result Confirmation is displayed in the **Command Log** view and the source appears in the **Library list** menu (*Objects* > *Library list*).

11.4.2 Adding a library to a library list from the Remote Systems view

Follow the subsequent steps to define your library list for your current RSE connection (session).

> **Note**
> When you close RDi or sign out, the lists created this way will be deleted. To save lists permanently, refer to Configuring a permanent library list on the next page.

Step 1 From the **Remote Systems** view, expand the connection for which you want to create a new library list.

Step 2 Expand the **Objects** menu.

Step 3 Right-click on **Library list**, then select **Add Library List Entry...**.

Result The **Add Library List Entry** dialog is displayed.

Step 4 Enter the name of the library to add to the list in the **Library** field, then click **OK**

Result The new list is created and appears in the **Library list** menu (*Objects* > *Library list*).

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

11.4.3 Configuring a permanent library list

Like a green screen session, adding a library the way described above will only create the list for the life of the active connection (session).

RDi enables you to assign a library list to a connection on a permanent basis so that it is always available, even after you close the session. Add a library as a property of the connection so every time the connection is activated the libraries will automatically become part of your session.

Step 1 From the **Remote Systems** view, right-click on the connection that contains the library.

Step 2 Select **Properties**.

Step 3 From the **Properties for [*Connection Name*]** dialog, select the **Subsystems** menu on the left.

Step 4 Enter the name of the library list to permanently add to the sessions in the **Library** field, then click **Add**.

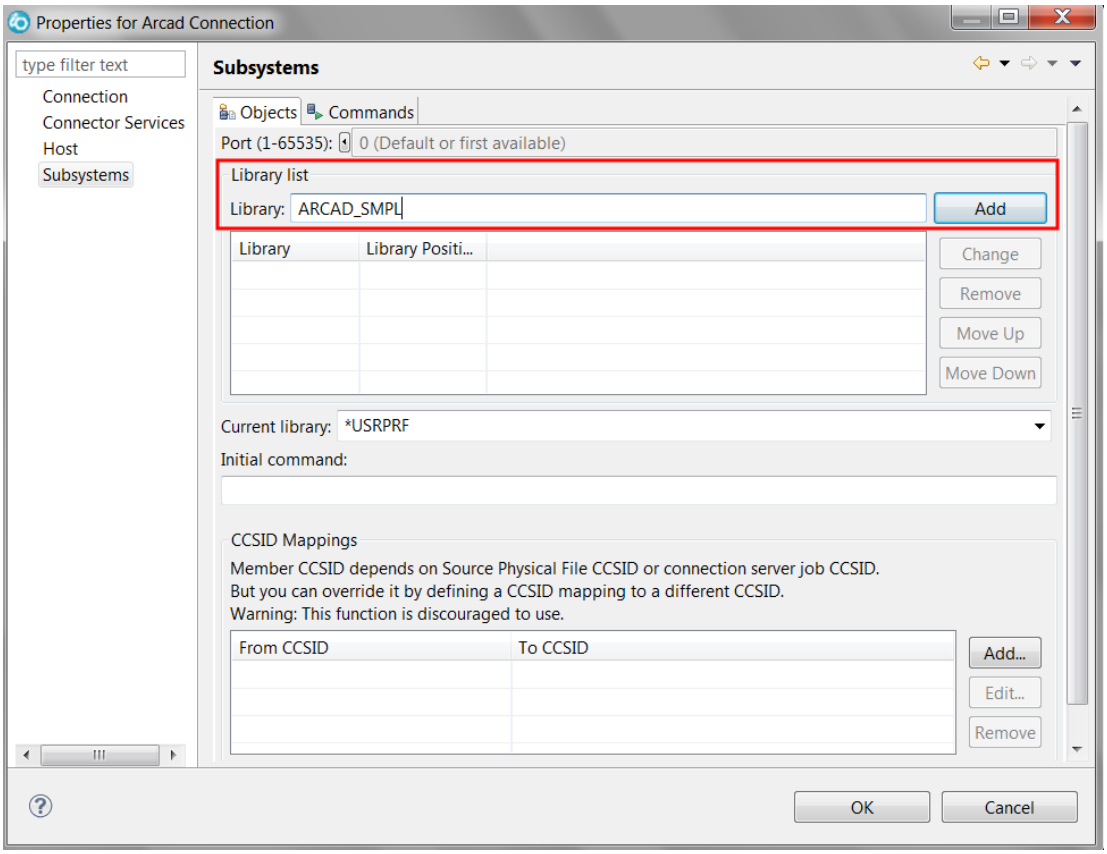


Figure 7: Add Library List to Session

Result The library is added to the table in the **Library list** section.

Step 5 Click **OK**.

Result RDi issues an `ADDLIBLE` command every time the connection is activated for each library listed in this table. This means the library list will automatically be prepared and ready for use immediately.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

12 Selecting source members

Chapter summary

The first step to convert source objects is to select the fully-qualified source member to convert (the library, the source file, the source type and the member name). ARCAD-Transformer RPG enables you to convert source members selected from both the **i Projects Navigator** and the **Remote Systems** views.

When the source member is selected, you can either transform it individually or add it to a conversion list to be transformed with other sources in a set.

> ***Reference***
> For more information about these actions, refer to Launching single-file conversions on page 85 and Launching mass conversions on page 94.

12.1 Selecting members from the i Projects Navigator

At least one IBM i Project with linked source members must be available to select a source member from this view.

Step 1 Open the **i Projects Navigator** (*Window > Show View > IBM i > i Projects Navigator*).

Step 2 Browse to and expand the source file containing the .rpg, .rpgle or .sqlrpgle source member(s).

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

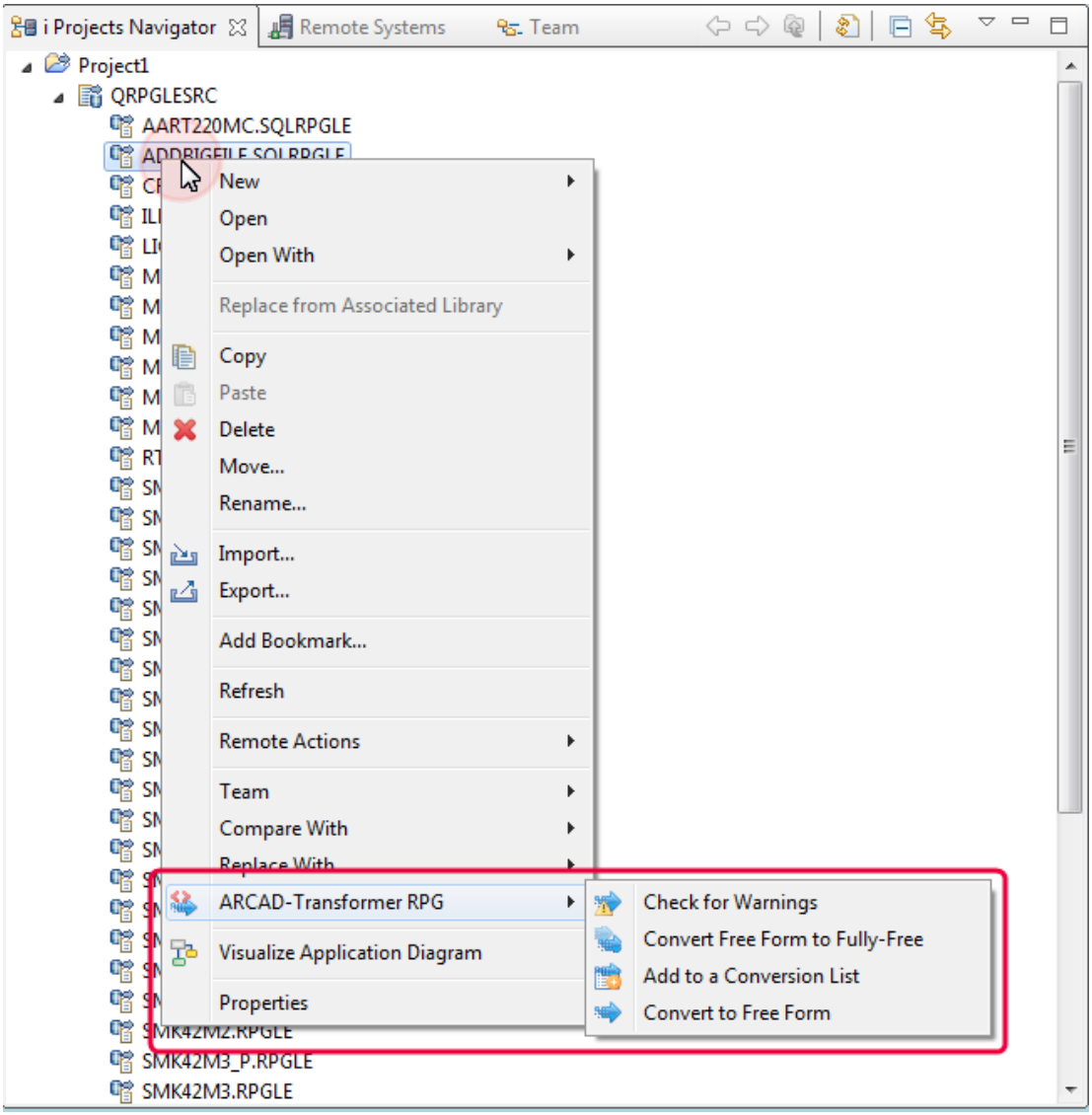


Figure 8: Selecting source members from the i Projects Navigator

12.2 Selecting members from the Remote Systems view

At least one IBM i server connection with linked source members must be available to select a source member from this view.

Step 1 Open the **Remote Systems** view (*Window* > *Show View* > *Remote Systems* > *Remote Systems*).

Step 2 Create a library and a member filter, if you don't have one already. See Preparing the environment on page 69.

Step 3 Browse to and expand the source file containing the .rpg, .rpgle or .sqlrpgle source member(s).

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

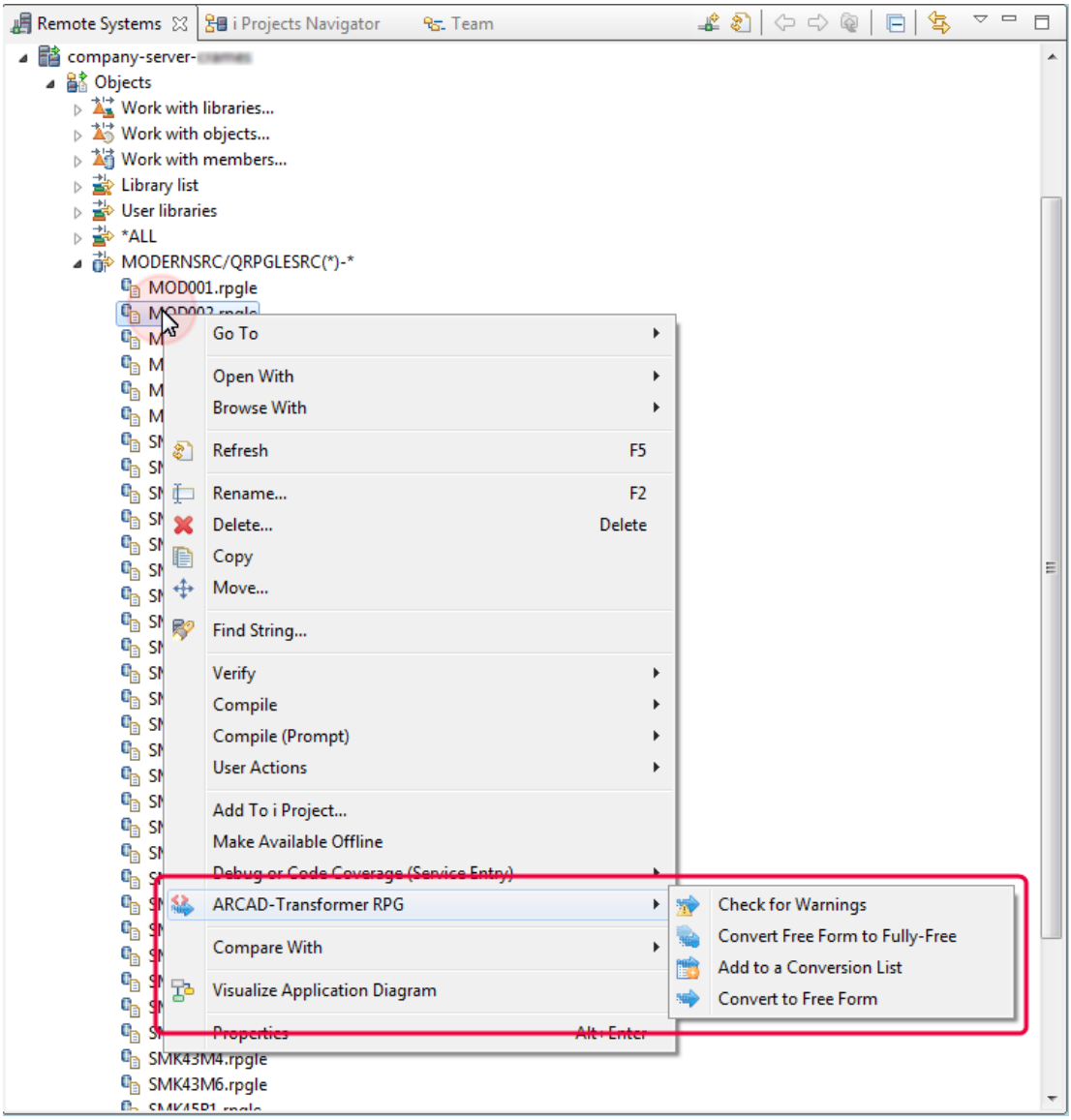


Figure 9: Selecting source members from the Remote Systems view

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

13 Resolving conversion warnings

Chapter summary

Before converting a source member, ARCAD-Transformer RPG checks the member for conversion warnings and allows you to decide what to do with each warning issued. This integrity check is intended to verify that the sources will convert correctly.

Checking a source member for conversion warnings audits the code and identifies the possible errors that could result from converting that source member.

> ***Example***
> Conversion warnings indicate any risk of numeric truncation or any risk due to the use of %Found or %Equal indicators.

If any conversion warnings are issued, a decision must be made regarding each warning. Once all the warnings have been addressed, the source member can be converted and the solution chosen for each warning will be carried out.

> **Note**
> When a source member is checked for warnings, the possible risks due to conversion are analyzed but no conversion unit is used during the process. The source member will not be converted.
>
> Change the Convert calculation specs (CVTCLCSPEC) parameter to change between running the conversion and checking for warnings.

13.1 Checking for conversion warnings

In RDi, conversion warnings can be checked from:

- a conversion list's editor,
- the **Conversion Warnings** view,
- the **Remote Systems** view, and
- the **i Projects Navigator**.

If the `ACVTRPGFRE` command is run from the command line, and if any of the warning option parameters have values other than *NO, conversion warnings will also be checked.

Follow the subsequent steps to check a source member for conversion warnings.

Step 1 Select one or multiple source members from the desired view or editor.

Step 2 Right-click on the source member(s) selected, then select **Check for Warnings**.

Result A wizard appears, displaying all the conversion options available.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Step 3 Define the necessary information in the **Conversion Options** and the **Advanced Options** pages of the wizard. Click **Next >** to access the second page.

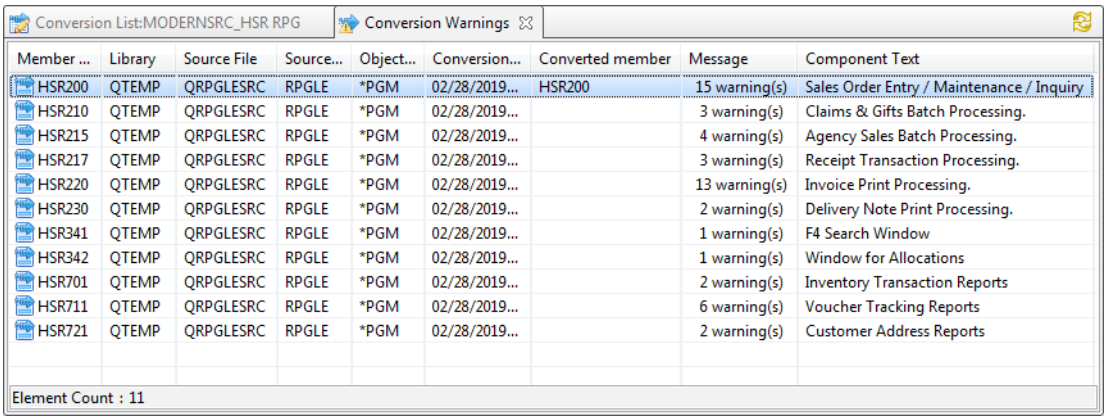
All the parameters are predefined with the defaults set in the **Free-Form Conversion Options** for the plug-in in the **Preferences** window.

> ***Reference***
> For more information about the parameters in the wizard and their preset default settings, refer to Managing the Free-Form conversion options on page 42.

Step 4 Click **Finish**.

Result Each selected source member is checked for warnings.

13.2 Accessing conversion warnings



Conversion List:MODERNSRC_HSR RPG | Conversion Warnings

Member ...	Library	Source File	Source...	Object...	Conversion...	Converted member	Message	Component Text
HSR200	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...	HSR200	15 warning(s)	Sales Order Entry / Maintenance / Inquiry
HSR210	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		3 warning(s)	Claims & Gifts Batch Processing.
HSR215	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		4 warning(s)	Agency Sales Batch Processing.
HSR217	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		3 warning(s)	Receipt Transaction Processing.
HSR220	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		13 warning(s)	Invoice Print Processing.
HSR230	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		2 warning(s)	Delivery Note Print Processing.
HSR341	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		1 warning(s)	F4 Search Window
HSR342	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		1 warning(s)	Window for Allocations
HSR701	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		2 warning(s)	Inventory Transaction Reports
HSR711	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		6 warning(s)	Voucher Tracking Reports
HSR721	QTEMP	QRPGLESRC	RPGLE	*PGM	02/28/2019...		2 warning(s)	Customer Address Reports

Element Count : 11

Figure 10: Conversion warnings

Conversion warnings are displayed in the **Conversion Warnings** view. This view is accessed from any perspective in RDi. Follow the subsequent steps to access the **Conversion Warnings** view.

Step 1 To open the **Show View** wizard, open the **Window** menu and select **Show View** > **Other...**.

Step 2 Expand the **ARCAD-Transformer RPG** folder and select **Conversion Warnings**.

> **Note**
> Views that are already opened are displayed in a lighter gray color.

Click **OK**.

Step 3 *[Optional]* If a connection has not already been made, a dialog is displayed and a connection must be selected or defined.

> ***Reference***
> For more information about connections, refer to Introduction to configuration on page 36.

Result The **Conversion Warnings** view opens. This view displays all the conversion warnings

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

detected, on a member-by-member basis for the members included in the connection. If a member has not been checked for warnings yet or did not return warnings, it is not displayed in this view. The **Message** column displays the number of warnings detected for each member.

Click any column header to reorder the list.

13.3 Resolving conversion warnings

Each individual conversion warning can, and often must be resolved. A decision is made regarding the warning and how it should be dealt with during the conversion.

Conversion warnings are resolved in the **Conversion Warning Resolution** view. There are two ways to access this view in RDi but, no matter the method, the view can be accessed from any perspective in RDi.

1. From the Window menu. When accessed from the **Window** menu, the **Conversion Warning Resolution** view displays all the conversion warnings detected for all the members included in the selected connection.
2. From the Conversion Warnings view or a conversion list editor. When accessed from the **Conversion Warnings** view or a conversion list's editor, the **Conversion Warning Resolution** view only displays the conversion warnings detected for the selected member.

13.3.1 From the Window menu

Follow the subsequent steps to access the **Conversion Warning Resolution** view from the Window menu.

Step 1 To open the **Show View** wizard, open the **Window** menu and select **Show View** > **Other...**.

Step 2 Expand the **ARCAD-Transformer RPG** folder and select **Conversion Warning Resolution**.

> **Note**
> Views that are already opened are displayed in a lighter gray color.

Click **OK**.

Step 3 [*Optional*] If a connection has not already been made, a dialog is displayed and a connection must be selected or defined.

> ***Reference***
> For more information about connections, refer to Introduction to configuration on page 36.

Result The **Conversion Warning Resolution** view opens. This view displays all the conversion warnings detected for all the members included in the connection. Each line corresponds to one conversion warning. If a member name is repeated through several lines, it means the member returned multiple conversion warnings.

Click any column header to reorder the list.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

13.3.2 From the Conversion Warnings view or a conversion list editor

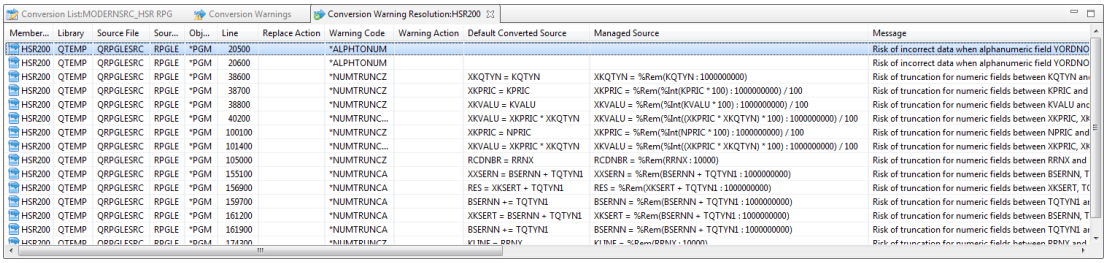


Figure 11: Conversion warning resolution - Warnings for a specific member

To access the **Conversion Warning Resolution** view from the **Conversion Warnings** view or a conversion list editor, either:

- double-click on a member in the **Conversion Warnings** view or in a conversion list's editor, or
- right-click on a member in the **Conversion Warnings** view or in a conversion list's editor, then select **Resolve Conversion Warnings**.

The view displays all the conversion warnings detected for the selected member. The warnings detected for other members cannot be displayed when the view is accessed from the **Conversion Warnings** view.

Click any column header to reorder the list.

13.4 Resolving conversion warnings

Conversion warnings can be resolved to determine how the warnings should be dealt with or if they should just be ignored. Conversion warnings are resolved from the **Conversion Warning Resolution** view.

In the **Conversion Warning Resolution** view:

- the **Default Converted Source** column displays, for each warning, the instruction that would have been used by default for the conversion and that is responsible for the conversion risk, and
- the **Managed Source** column displays a converted instruction suggested by ARCAD-Transformer RPG to replace the default one and prevent any conversion risk.

Follow the subsequent steps to update a conversion warning.

Step 1 To open the **Update Resolution** wizard, either:

- double-click on a warning in the **Conversion Warning Resolution** view, or
- select one or multiple warnings in the **Conversion Warning Resolution** view, then right-click on the selection and select **Update Resolution**.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

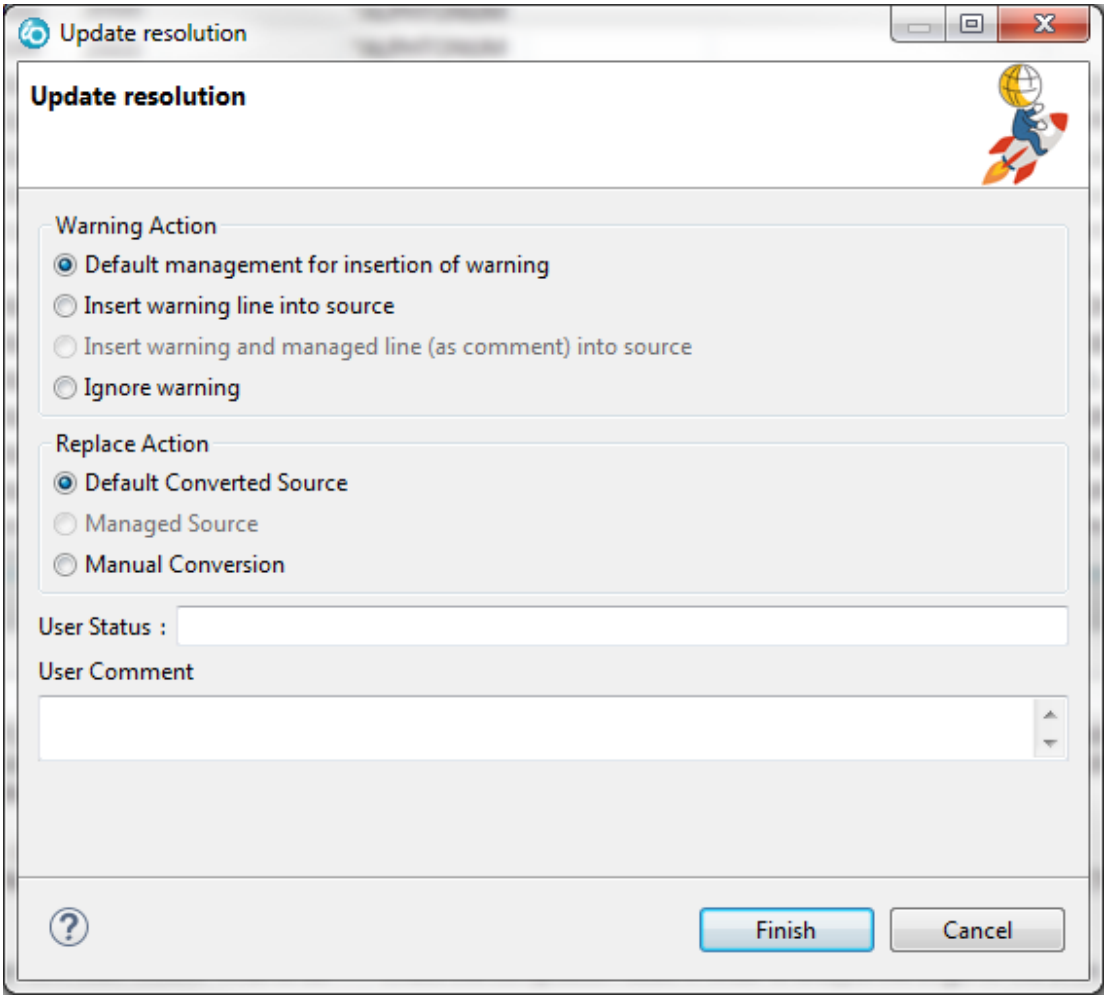


Figure 12: Resolve conversion warnings

Step 2 Define how the warning is written into the converted source or if it should be ignored by checking a box in the **Warning Action** section.

Check-box	Description
Default management for insertion of warning	This default depends on the nature of the conversion warning (for example *CHECKIND or *NUMTRUNCB) and the preference set for this type of conversion warning. ***Reference*** For more information about these preset preferences, refer to Managing the Free-Form conversion options on page 42.
Insert warning line into source	A warning is added as a comment right after the instruction in the converted source member. **Note** Checking this option is the same as

Table 67: Updating conversion warnings - Warning Action

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Check-box	Description
	selecting the *WNG1 value for the warning conversion options.
Insert warning and managed line (as comment) into source	A warning is added as a comment right after the instruction in the converted source member. The warning also includes the instruction suggested for the managed source. **Note** Checking this option is the same as selecting the *WNG2 value for the warning conversion options.
Ignore warning	The conversion warning is ignored and no comment is added into the converted source member.

Table 67: Updating conversion warnings - Warning Action

Step 3 Define how the instruction is modified in the converted source to deal with the warning by checking a box in the **Replace Action** section.

Check-box	Description
Default Converted Source	The instruction specified in the **Default Converted Source** column in the view will be used in the converted source member. This instruction corresponds to the converted instruction that would have been used automatically if it did not represent a conversion risk.
Managed Source	The instruction specified in the **Managed Source** column in the view will be used in the converted source member. This instruction is the converted instruction suggested by ARCAD-Transformer RPG to replace the default converted instruction responsible for the conversion warning.
Manual Conversion	Users must edit and manually change the instruction in the converted source member. In the mean time, the instruction specified in the **Default Converted Source** column in the view will be used.

Table 68: Updating conversion warnings - Replace Action

Step 4 Define a **User Status** and a **User Comment** if needed. The status and comments given to a warning are for informational purposes only and are used to tag or categorize warnings. Any value entered in these fields will be displayed in the **Conversion Warning Resolution** view, so they can be used to sort or filter a list of warnings.

Step 5 Click **OK**.

Result The conversion warning selected is updated according to the options defined in the wizard. When you launch the conversion, the defined actions will be taken for each warning.

13.5 Viewing warnings in-line in source members

Browsing a source member, whether in its original or converted form, makes it possible to display the conversion warnings in context without editing the member. A source member can be browsed from both the **Conversion Warnings** and the **Conversion Warning Resolution** views.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Follow the subsequent steps to browse an original or a converted source member.

Step 1 Right-click either on a member in the **Conversion Warnings** view, or on a conversion warning in the **Conversion Warning Resolution** view.

Step 2 Select **View Converted Source Member** to display the member as it would be after its conversion.

Result The source member browser opens. The conversion warnings issued for the source member are highlighted to find them easily.

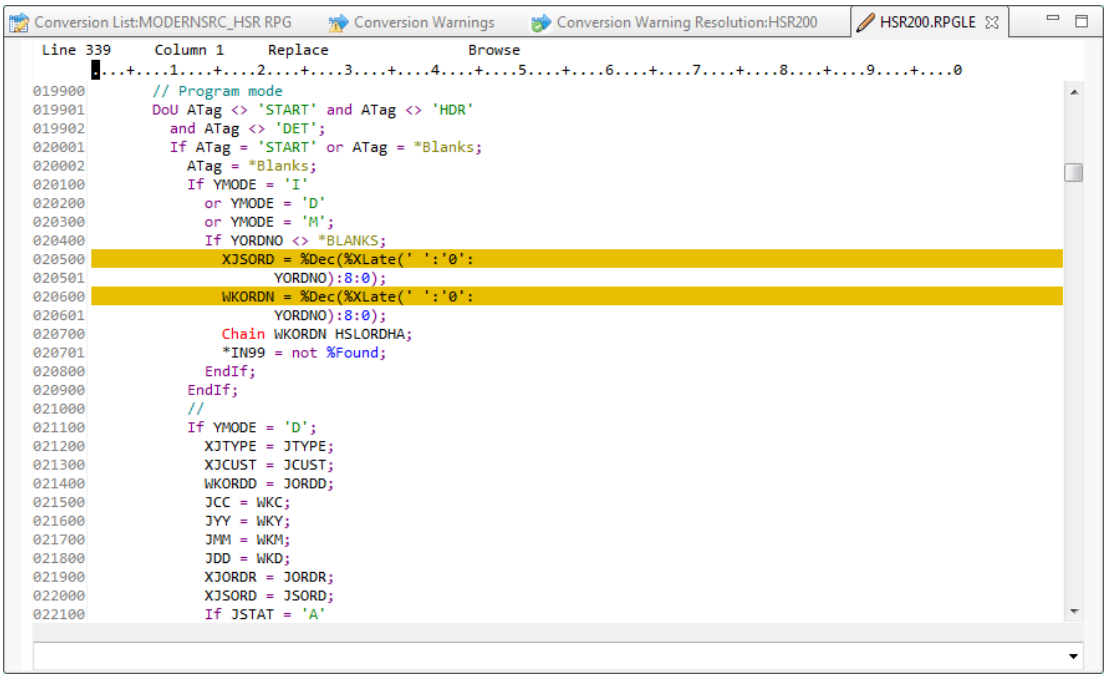


Figure 13: Conversion warnings in-line in the converted source

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

14 Launching single-file conversions

Chapter summary

Conversions can be done on individual member or on groups of members saved in a conversion list. This chapter contains information about preparing and executing the conversion of a single member.

> ***Reference***
> For more information about converting multiple members in a list, refer to Working with conversion lists on page 91.

14.1 Executing a single-file conversion

> **Note**
> Values entered in the **Converted Source Member Properties** wizard are stored as preferences.

Step 1 Select the source member to convert. See Selecting source members on page 75.

Step 2 Right-click and from the **ARCAD-Transformer RPG** menu, select **Convert to Free-Form**.

> **Note**
> If you select **Convert Free-Form to Fully-Free Form**, refer to Converting Free to Fully-Free Form on page 22

Step 3 Define the new source file in the **Converted Source Member Properties** wizard.

The following table contains a link to a complete description of each parameter defined in this wizard and the values allowed for each parameter.

Parameter	Sub-parameter name	Values
Source File (SRCFILE) (uneditable)	Library*	*Name*, ***LIBL**, *CURLIB
	Source File*	*Name*, **QRPGLESRC**
Source Member (SRCMBR)* (uneditable)		*Name*
Source Type (SRCTYPE) (uneditable)		**RPGLE**, SQLRPGLE
Object Type (OBJTYPE)		***PGM**, *MODULE, *NONE
Convert calculation specs (CVTCLCSPEC)		[*Optional*] *NO, *EXTFACT2, ***FREE**, *CHECK, *FREECHECK
Convert declaration specs (CVTDCLSPEC)		***YES**, *NO

Table 69: Converted Source Member Properties

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Parameter	Sub-parameter name	Values
Destination Source File (TOSRCFILE)	Library	[*Optional*] *Name*
	Source File	[*Optional*] Single values: ***NONE**, *FROMFILE Other values: *Qualified object name*
	Member Name (TOSRCMBR)	[*Optional*]*Name*, ***FROMMBR**
Source Line Date (SRCDATE)		[*Optional*] ***CURRENT**, *ZERO, *KEEP

Table 69: Converted Source Member Properties

1. The **Original Source Member** information.

Source File (SRCFILE)

This is the library and source file of the source member to analyze. These fields are automatically filled in with the information corresponding to the selected source member.

The possible values for the name of the library are:

Value	Description
***LIBL**	The object is searched for in the list of libraries.
*CURLIB	The specified object is searched for in the current library for the job. If no *CURLIB was specified, QGPL is used by default.
Library name	Indicate the name of the library containing the object.

Table 70: Source File (SRCFILE) library parameters

The possible values for the source file name are **QRPGLESRC**, the same value as the source file of the member to convert or nothing.

Source Member (SRCMBR)

The name of the source member to analyze. This field is automatically filled in with the information corresponding to the selected source member.

The source member must be RPGLE or SQLRPGLE and it should be a main source. The source may be a /COPY clause if you do *not* convert calculation specifications.

> **Note**
> You cannot access the command prompt page if the selected source member is RPG source member type. See Prompting the ACVTRPGFRE command on page 90.

Source Type (SRCTYPE)

The syntax for RPGLE or SQLRPGLE C specifications. This field is automatically filled in with the information corresponding to the selected source member.

Value	Description
RPGLE	The source contains RPGLE instructions without embedded SQL.
SQLRPGLE	The source contains RPGLE instructions and embedded SQL.

Table 71: Source Type (SRCTYPE) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Object Type (OBJTYPE)

To convert calculation specifications (C), this parameter indicates whether the object directly created from this source is a ***MODULE** or ***PGM**. This information is required when converting "*ENTRY PLIST ..." to Free syntax.

However, to convert declaration specifications only, the value ***NONE** allows the normal preliminary compilation to be omitted (useful particularly for COPY clauses).

Value	Description
***PGM**	For Free, the prototype for the input parameters of this program will have the **EXTPGM** keyword.
*MODULE	For Free, the prototype for the input parameters of this module will not have the **EXTPGM** keyword.
*NONE	No object type is associated with this source; this value can only be used if you are converting only the declaration specifications (H, F, D, P). It is not allowed if you want to convert the calculation specifications (C). **Important!** If the **Object Type** value is ***NONE**, the Convert calculation specs (CVTCLCSPEC) must be set to ***NO**. A message displays when the type is changed and the correct settings are automatically defined.

Table 72: Object Type (OBJTYPE) parameters

2. Define the specifications to convert.

Convert calculation specs (CVTCLCSPEC)

Define whether or not you want to convert calculation specifications (C).

Value	Description
*NO	No RPGLE syntax conversion is performed for calculation specifications (C).
*EXTFACT2	When possible, old codes operations using the factor 1, factor 2 and result are converted to instructions using extended factor 2.
***FREE**	All calculation instructions RPGLE (with some exceptions) are converted into Free syntax.
*CHECK	The source to be converted is analyzed to detect potential risks of truncation for numeric fields, but no conversion is performed. The conversion units count (linked to the product license) is not incremented. Any warnings issued can be viewed and managed before the official conversion of the source. ***Reference***

Table 73: Convert calculation specs. (CVTCLCSPEC) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Value	Description
	For more information, refer to Resolving conversion warnings on page 78.
*FREECHECK	The source to be converted is only officially converted if there are no warnings issued, so no conversion units are used in case of warnings. If at least one warning is issued indicating a risk of truncation for numeric fields, the conversion is not performed.

Table 73: Convert calculation specs. (CVTCLCSPEC) parameters

Convert declaration specs (CVTDLCSPEC)

Define whether or not to convert declaration specifications (D).

> ***Reference***
> For more information about D specs, refer to Convert declaration specs (CVTDCLSPEC).

3. Define the new **Converted Source Member**.

Destination Source File (TOSRCFILE)

The name of the destination library and source file that will house the converted source member.

> **Important!**
> You must have sufficient rights for the destination source file.

If the destination **Library** is QTEMP, the source file will be created with the same attributes as the original source file.

If the destination library is not QTEMP, the source file must already exist.

> ***Reference***
> Refer to the Authorize QTEMP in batch (BATCHQTEMP) parameter to authorize QTEMP in batch mode.

> **Note**
> The Source File and the Memer Name cannot be the same.

The possible values for the **Source File** name are:

Value	Description
***NONE**	No source member is generated as output; in this case, remember to set the Clean Modified Lines (CLRFRMCHG) parameter to ***NO** in order to see the new source lines in the *AARFCHSF1* file.
*FROMFILE	The destination library and source file are the same as the source file

Table 74: Destination Source File (TOSRCFILE) parameters

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Value	Description
	of the member to be converted.
Name of the source file	Define a name and a library for the destination source file.

Table 74: Destination Source File (TOSRCFILE) parameters

Member Name (TOSRCMBR)

Define the destination source member, if you selected a specific source file.

> **Note**
> The Source File and the Member Name cannot be the same.

Value	Description
***FROMMBR**	The destination source member has the same name as the source member analyzed.
Name of the member	Enter the member name receiving the converted source.

Table 75: Member Name (TOSRCMBR) parameters

> **Important!**
> If you select an existing source member, you must choose whether or not to **Replace the Existing Member** or create a copy.

Step 4 Define the conversion options.

Further parameters are found on the second and third pages of the wizard. Click **Next** to access these pages. These are the conversion options which can also be managed in the preferences menu.

> ***Reference***
> For more information about the conversion options, refer to Managing the Free-Form conversion options on page 42.

Step 5 Click **Finish** to start the conversion or click **Next >** to prompt the command manually.

> ***Reference***
> For more information about manually running the command, refer to Prompting the ACVTRPGFRE command on the next page.

Result The conversion is launched.

When the conversion is complete, if it was successful, you can open the new source member. If unsuccessful, an error message will be displayed giving details of the error(s).

> ***Reference***
> If the conversion fails and you need help understanding the error, refer to Troubleshooting on page 100.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

For more information about consulting converted content, refer to Understanding conversion results on page 96.

14.2 Prompting the ACVTRPGFRE command

ARCAD-Transformer RPG offers a way to prompt and manipulate the `ACVTRPGFRE` command just as it would be in a 5250 emulator. IBM i users will recognize the parameters in the command prompt menu. However, prompting the command manually accomplishes the same task as running the conversion via the wizard.

Follow the subsequent steps to prompt the command.

Note
Values entered in the command prompter are not stored as preferences.

Step 1 From the third page of the **Converted Source Member Properties** wizard, click the **Prompt the command** button to display the RDi command prompter.

Step 2 Enter the conversion options.

The values displayed in the command prompter are those defined on the first two pages of the wizard.

Reference
For more information about the options available, refer to Executing a single-file conversion on page 85 and Managing the Free-Form conversion options on page 42.

There is one parameter available in the command prompt that is not available in the wizard:

Authorize QTEMP in batch (BATCHQTEMP)

This parameter allows the authorization of an output source file located in QTEMP for the converted member, even if the job is executed in batch mode.

Value	Description
***NO**	By default, QTEMP is not allowed for the output source file library, when the conversion is done in batch. This especially avoids making a conversion from a GUI for which the result is not accessible.
*YES	Indicate ***YES** if the `ACVTRPGFRE` command is included in a macro that retrieves the source from QTEMP to integrate it into a development library.

Table 76: Authorize QTEMP in batch (BATCHQTEMP) parameters

Step 3 Click **OK** to display the command string.

Step 4 Click **Finish** to execute the command and launch the conversion.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

15 Working with conversion lists

In order to launch a conversion on a number of items that should share the same output library and source, you must create a conversion list. This section contains information about preparing conversion lists.

A conversion item is a reference to the conversion of a source member. It contains the following information:

- The fully-qualified source member to convert (the library, the source file, the source type and the member name.

> ***Reference***
> Managing the Free-Form conversion options on page 42 and Managing the default RPG to RPGLE conversion options on page 67

- The object type used during the conversion.
- The name of the converted source member.
- The date, the status and the related message of the previous conversion.

You can group some conversion items into a conversion list if you want them to share the same IBM i connection as well as a unique source file where all of the converted source members will be saved.

All the source member compilations will run in the same job for a conversion list and all the results will be located in the same place. These shared values are stored within the list header which is defined when a list is created and can be managed in the **Conversion List** editor.

15.1 Creating conversion lists

There are two ways to create a new conversion list. You can create one when adding an individual file to a list and you can create empty lists from the **Conversion List** view.

15.1.1 Create conversion lists with new sources

Follow the subsequent steps to create a conversion list from the **Add to a conversion list** wizard.

Step 1 Select a source member to add to a conversion list (see Selecting source members on page 75).

Step 2 Right-click and from the **ARCAD-Transformer RPG** menu, select **Add to a Conversion List**.

Step 3 Click **Add...** at the bottom of the wizard to create a new list.

Result The **Create a conversion list** wizard is displayed.

Step 4 Define the required information for the list.

> ***Reference***
> For more information about these fields, refer to Editing conversion lists on the next page.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Step 5 Click **Finish.**

Result The new conversion list is created and includes the selected source(s). It is available in the **Conversion List** view.

15.1.2 Create empty conversion lists

Follow the subsequent steps to create a conversion list from the **Conversion List** view.

Step 1 Click the Create a new conversion list icon in the toolbar or right-click anywhere in the view and select **Create a conversion list**.

Result The **Create a conversion list** wizard is displayed.

Step 2 Define the required information for the list.

> ***Reference***
> For more information about these fields, refer to Editing conversion lists below.

Step 3 Click **Finish.**

Result The new conversion list is created and available in the **Conversion List** view.

15.2 Populating conversion lists

Follow the subsequent steps to add source members to an existing conversion list.

Step 1 Select a source member to add to a conversion list (see Selecting source members on page 75).

Step 2 Right-click and from the **ARCAD-Transformer RPG** menu, select **Add to a conversion list**.

Step 3 Select an existing conversion list from the **Select a conversion list** wizard.

Step 4 Click **Finish**.

Result The source file is added to the conversion list.

15.3 Editing conversion lists

Conversion lists are managed in the **Conversion List** view. Double-clicking on a list opens the **Conversion List** editor where the list of all of the source files included in it can be found.

The defined properties in the list header can be modified in the editor.

Parameter	Definition
Connection Name *	Click the browse icon to change the server in which the list should be stored.
List Name*	Enter a unique name for the list.
List Description	Enter a short description of the list to help you identify it.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

The **Target Information** section contains parameters that define the shared output source of all the files in the list.

Parameter	Definition
Library*	Enter the name of the shared library that contains the target source file.
Source File*	Enter the name of the source file where the converted members will be saved.

The list of source files contains all the key information about each source, including the converted member's object type and the date the conversion was made.

15.4 Deleting conversion lists

To delete a conversion list either select the list(s) to delete from the **Conversion List** view and click the delete icon or right-click on the list and select **Delete conversion list**.

Click **OK** to confirm the permanent removal of the selected list(s).

15.5 Editing source members

ARCAD-Transformer RPG allows you to edit original and converted source members.

15.5.1 Edit original source members

Follow the subsequent steps to edit an original source member.

Step 1 Open a conversion list in the editor (see Editing conversion lists on the previous page).

Step 2 Right-click on the member included in the original source file you want to edit.

Step 3 Select **Edit original source member** from the context menu.

Result The source editor opens.

15.5.2 Edit converted source members

Follow the subsequent steps to edit a converted source member.

Step 1 Open a conversion list in the editor (see Editing conversion lists on the previous page).

Step 2 Right-click on the member included in the converted source file you want to edit.

Step 3 Select **Edit converted source member** from the context menu.

Result The source editor opens.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

16 Launching mass conversions

Chapter summary

Conversions can be done on individual files or on groups of files saved in a conversion list. This chapter contains information about preparing and executing mass conversions on items contained in conversion lists.

> ***Reference***
> For more information about converting individual files, refer to Launching single-file conversions on page 85.

16.1 Defining object types

Defining an object type is mandatory to convert a source member because it determines the way this source member will be compiled.

> **Important!**
> The object type is defined in the **Conversion Wizard** when executing a single-file conversion, however, because the mass-conversion wizard only allows you to enter common conversion options, this value must be defined before-hand for mass conversions.

Follow the subsequent steps to define the object type for members in preparation for mass-conversion.

Step 1 Open a conversion list in the editor (see Editing conversion lists on page 92).

Step 2 Right-click on the member(s) for which you need to change the object type.

Step 3 Select **Update Object Type** from the context menu.

Step 4 Select the object type from the drop-down list.

Step 5 Click **Finish** to update the type.

Step 6 Save the conversion list.

Result The selected object type appears in the **Object Type** column in the conversion list.

16.2 Converting members in a list

Conversion lists enable you to convert members that are saved in a list and ensure that the converted members share the same library and source. You can either convert all of the members in a list at the same time, or you can select specific members to convert together.

Follow the subsequent steps to launch a mass-conversion.

Step 1 Select the member(s) to convert in the conversion list.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Important!
Check that an object type has been defined for each of the selected conversion items. Refer to Defining object types on the previous page.

Step 2 Right-click on the member(s) and select **Convert**. Then select either **Convert the selected items** or **Convert all**.

Step 3 Enter the necessary conversion options in the **Conversion Wizard**.

Note
Because the **Convert Decl. Specification** option is not usable if the **Object Type** is ***NONE**, this option will be automatically deactivated for all the members for which those options are in conflict.

A message will be displayed in the log to inform you of this modification.

Reference
For more information about the parameters in the wizard, refer to Executing a single-file conversion on page 85 and Managing the Free-Form conversion options on page 42.

Step 4 Click **Finish**.

Result The conversion is launched.

When the conversion is complete, if it was successful, you can open the new source member. If unsuccessful, an error message will be displayed giving details of the error(s).

Reference
If the conversion fails and you need help understanding the error, refer to Troubleshooting on page 100.

For more information about consulting converted content, refer to Understanding conversion results on page 96.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

17 Understanding conversion results

Chapter summary

Each successful conversion uses one conversion unit and each ARCAD-Transformer RPG license contains a predefined number of conversion units. Licenses do not track which source members have already been converted, therefore running a conversion program on a file once, then modifying something and running the same program again counts as two conversions.

17.1 Comparing original and converted source members

In addition to the log displayed after a conversion, you can compare the original source member to the converted version to see the results.

If you select **OK** when prompted to open the converted source member after a successful conversion, the new source opens in the editor. Open the original member and put them side by side to compare the transformation.

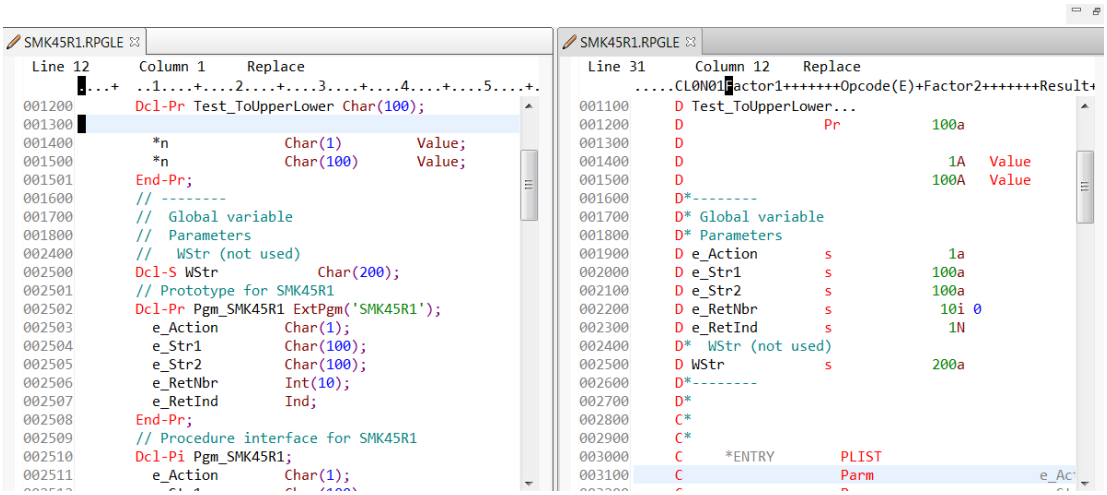


Figure 14: Comparing original and modernized sources

17.2 Conversion status

To keep a record of your actions, the results of the most recent conversion made to any item(s) are stored in the conversion item and displayed in the conversion list.

The result information consists of four values displayed in the table:

Value	Description
Status	This is the current status of the item in the conversion process. The possible values are: No Icon = No conversion has been made.

Table 77: The conversion statuses

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Value	Description
	✔ = The conversion succeeded. ✔ with Messages = The conversion succeeded but there are some warnings. ✘ = The conversion failed.
Conversion Date	The date of the last conversion attempt.
Conversion Member	The name of the converted member. Generally this is the same as the original name but the location of the source member is different.
Message	This is a summary of the conversion process. The entire message is displayed at the bottom of the editor when a conversion item is selected.

Table 77: The conversion statuses

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Appendices

Copybooks in ARCAD-Transformer RPG

At the moment when ARCAD-Transformer RPG automatically converts a copybook, it only touches the comment lines and leaves all of the active lines of code intact. This is due to the fact that it uses the RPG compiler to generate the necessary field references. Since a copybook cannot be compiled by itself, ARCAD-Transformer RPG bypasses the conversion of copybooks because of the inherent inability for successful stand-alone compilation.

What has been suggested to convert the copybook would be to take the copybook and rename it. Then put the copybook along with all of the members that are needed to make up the entire executable, i.e., the program and all the satellite copybooks that are called within it, into a member. Once that conversion of the whole program complex has been successfully undertaken, the portion of code which represented only the copybook could be extracted manually from the overall code and put into its own member to then become a "converted" copybook.

However, the possibility that the copybook contains subroutines, lines of code that initiate with the BEGSR command and terminate with the ENDSB command, may exist. Since ARCAD-Transformer RPG offers an option for converting subroutines to procedures, if the copybook contains any of these subroutines and the conversion is set up to process these routines into external procedures, the original copybook containing the subroutine(s) will be replaced by a member containing CALLP commands linked to the procedures, which would then exist as procedures carrying the name of the subroutines. Should there be any other programs that referenced the original copybook, those programs would no longer function because the copybook could no longer supply the original code for the required subroutines. This is the primary reason why ARCAD-Transformer RPG does not convert the code in copybooks.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Troubleshooting

CPF9801 – The object xxx in library yyy type *FILE not found

Problem

Before the conversion process occurs, ARCAD-Transformer RPG validates the existence of the destination library and source physical file. If it is not found an error is reported.

Resolution

Verify the destination spelling and retry your conversion.

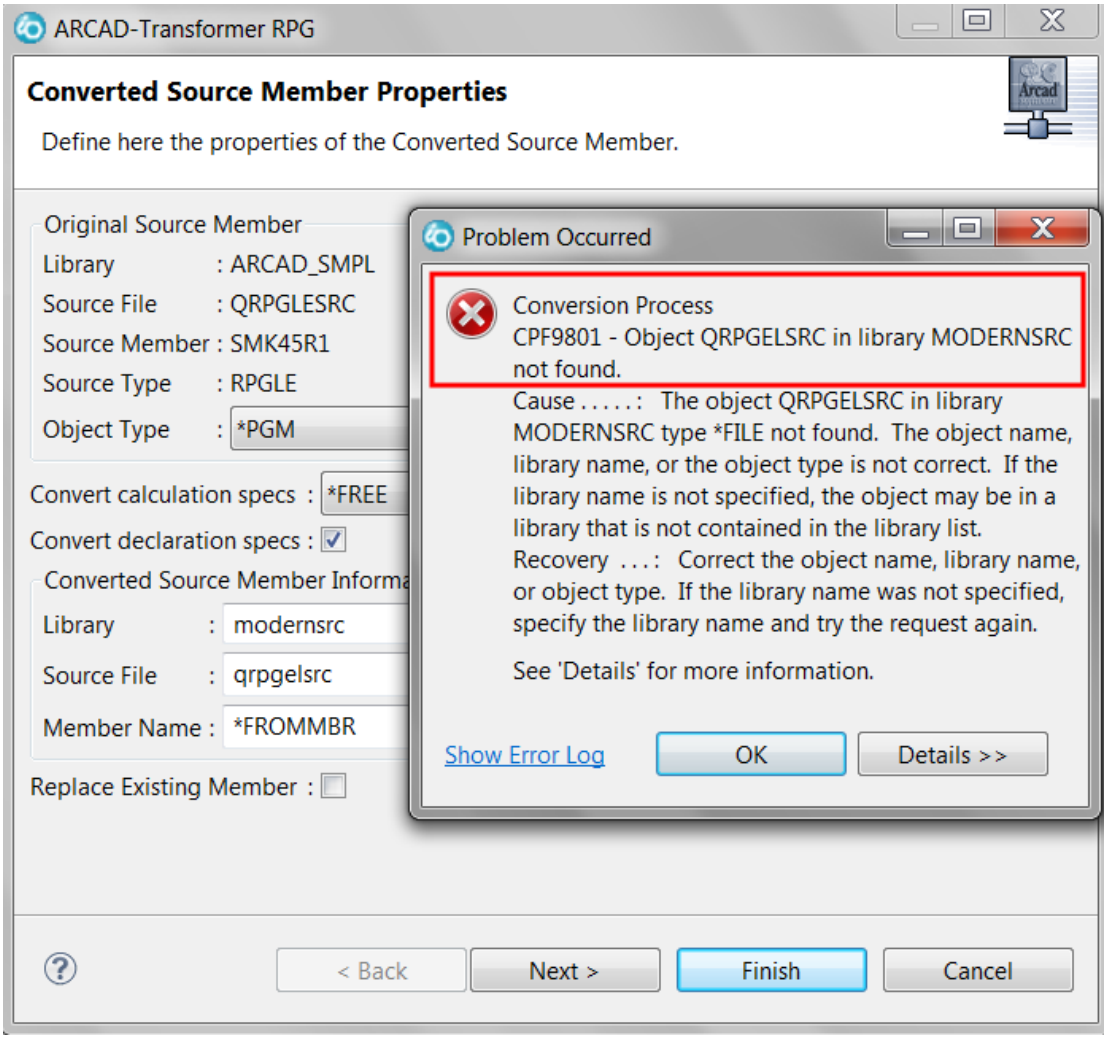


Figure 15: Problem Occurred CPF9801

MSG3542 – Member xxx already exists in source file yyy/zzz

Problem

You are attempting to convert a source member and the destination library / source physical file and member already exist.

1. Resolution 1
   Choose a destination with a member name for your converted source that does not already exist.
2. Resolution 2

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

If you intend to use your destination and member name, select the **Replace Existing Member** checkbox.

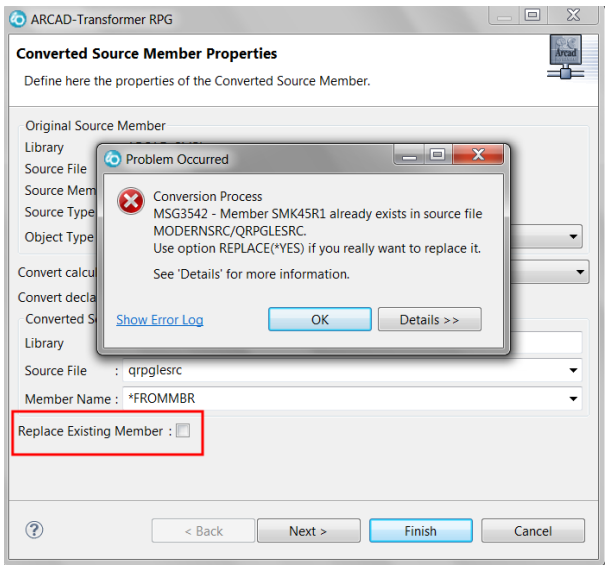


Figure 16: Problem Occurred MSG3542

MSG3579 – The CCSID of the output source file xxx is different from the CCSID of the entry source file yyy

Problem

A mismatch of CCSIDs can potentially cause misconverted or non-compilable source code.

1. Resolution 1
   Define a different target destination to contain your converted source code.
2. Resolution 2
   Change the CCSID of the target destination to match your source member that is being converted.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

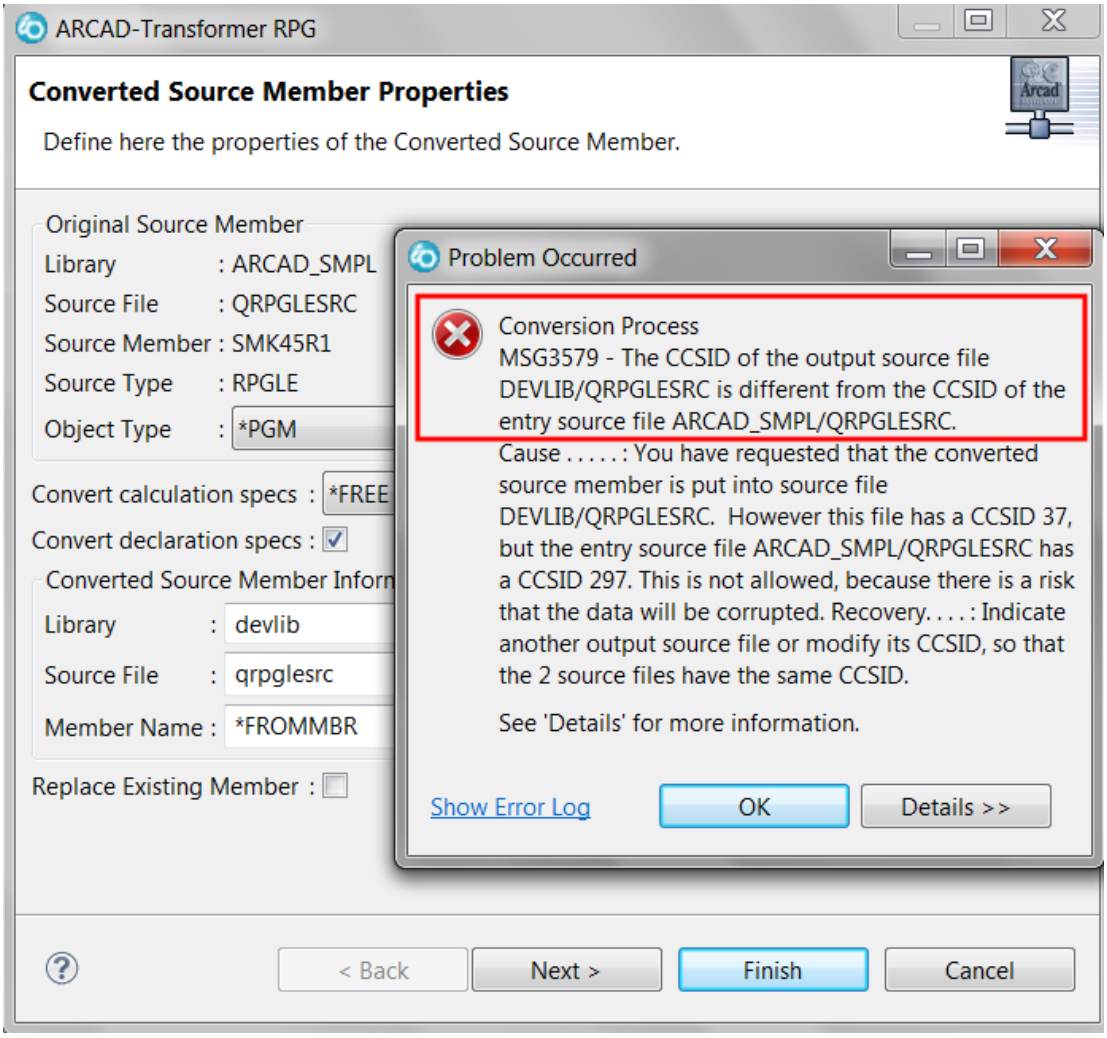


Figure 17: Problem Occurred MSG3579

In this particular example DEVLIB/QRPGLESRC has a CCSID of 37 and ARCAD_SMPL/QRPGLESRC is 237.

If you want to change the CCSID of DEVLIB/QRPGLESRC you will issue a CHGPF command to have it become CCSID 237:

MGR3866 – Error during preparation XRef calculation for xxx: impossible to convert to Free

Problem

A cross reference must be generated in order for Transformer RPG to convert your source code.

The most common cause is that the runtime environment is not prepared properly. See below for specific error messages.

There are several ways to identify why a source member did not convert.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

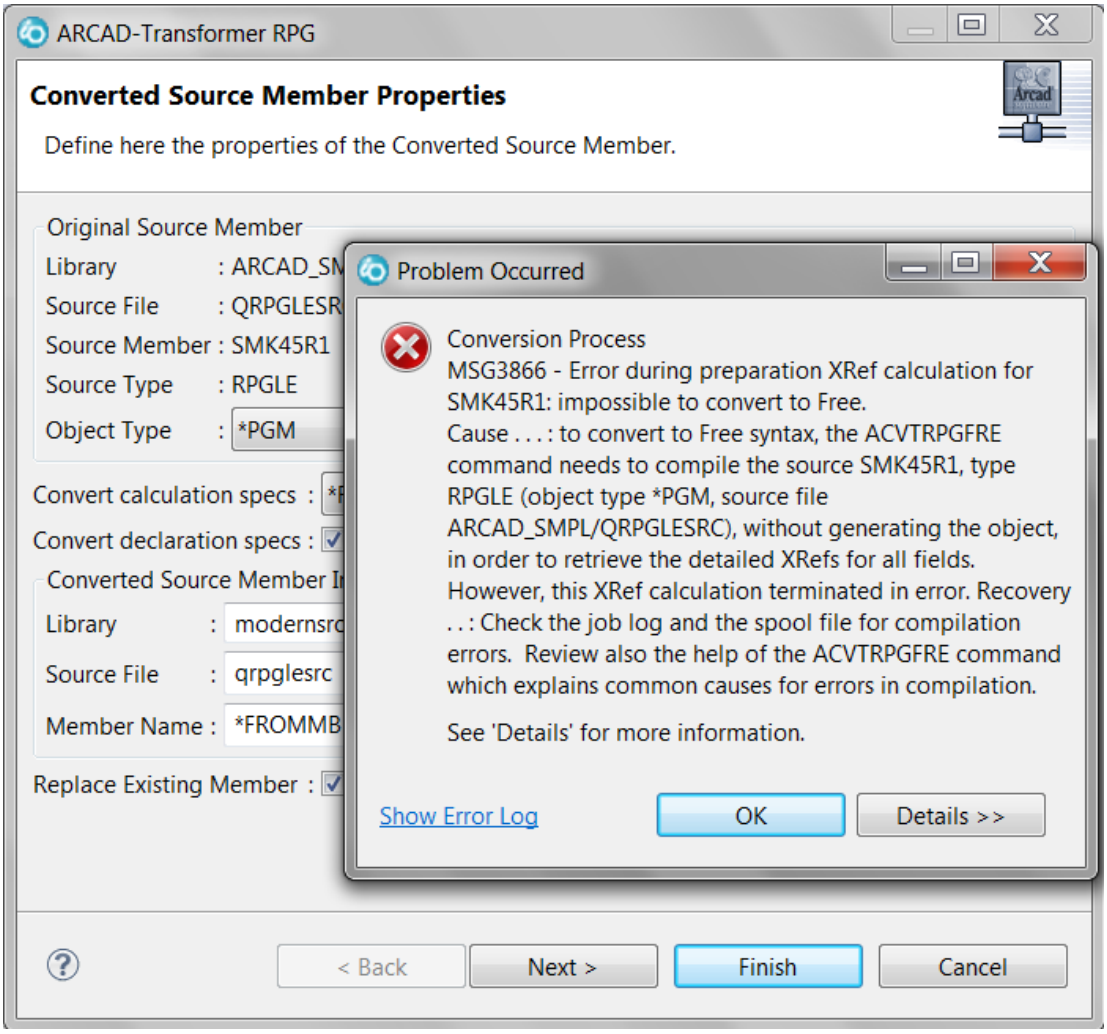


Figure 18: Problem Occurred MSG3866

1. Resolution 1
   Incorrect library list – see Adding libraries to library lists on page 72 and then retry.

2. Resolution 2
   Review the **Error Log** view for error messages. If the **Error Log** view is not present on your perspective a quick way to restore it is using RDi's Quick Access. Quick Access is located near the tip of the perspective. In this space type **Error Log**. When the **Error Log** view appears in the search results click on it.

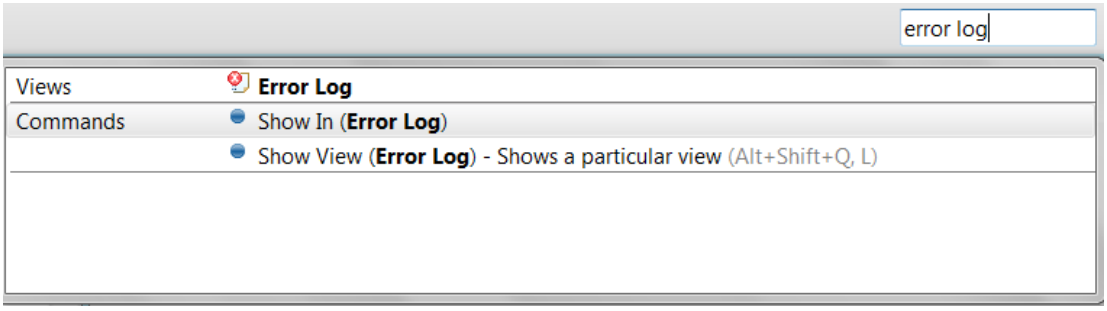


Figure 19: Display Error Log view

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

Look for the most recent entries in the error log. These will show in detail the causes for the conversion problems.

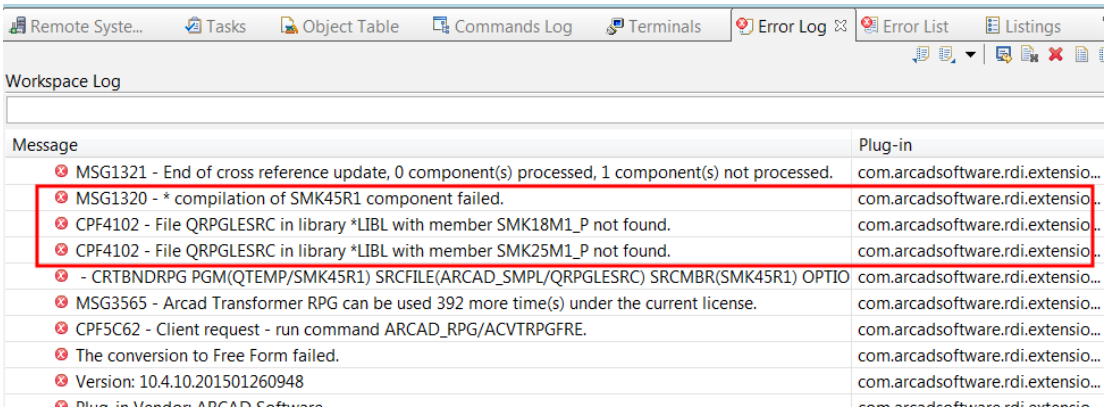


Figure 20: Error Log entries

3. Resolution 3
   Read the spool file that is created during the conversion process. The option to view spooled files is presented automatically if a single-file conversion fails:

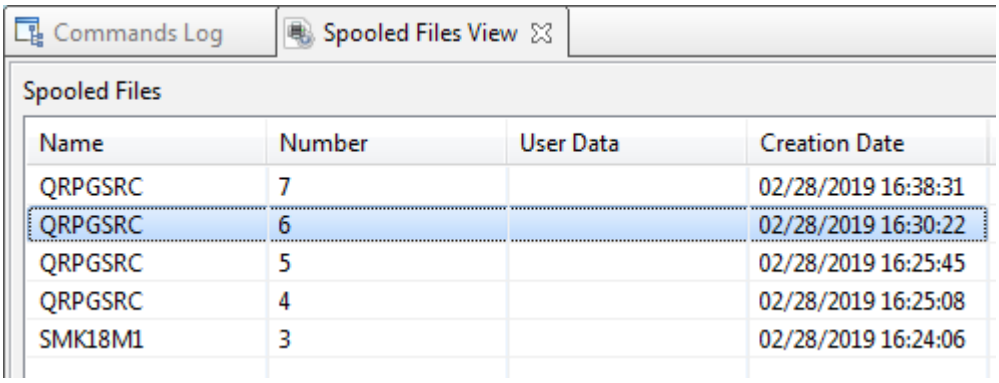


Figure 21: Spooled Files view

Spooled files can also be accessed in RDi from the **Spooled Files** filter.

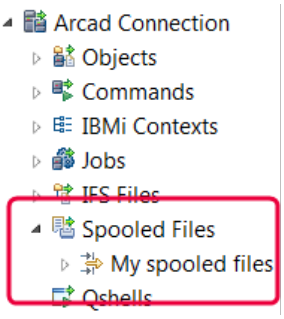


Figure 22: My spooled files

a. Open **My spooled files** and see *all spooled files for your user ID. If there are not a lot of entries this is a reasonable approach. For ARCAD-Transformer RPG, the printer file will be the same name as the program being converted. In this example we have been using source member SMK45R1. Right-click on **Spooled Files** and select New to create a new filter. Next, select **Spooled File Filter...**.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

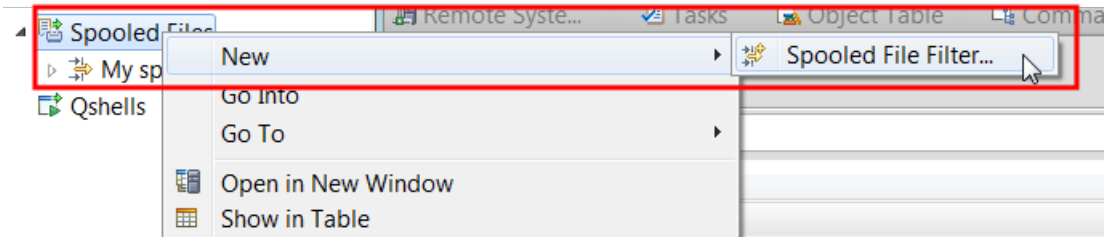


Figure 23: Spooled File filter

b. Leave the user ID as *CURRENT. Type the program name in the **Spooled File Name** field. Then click **Next**.

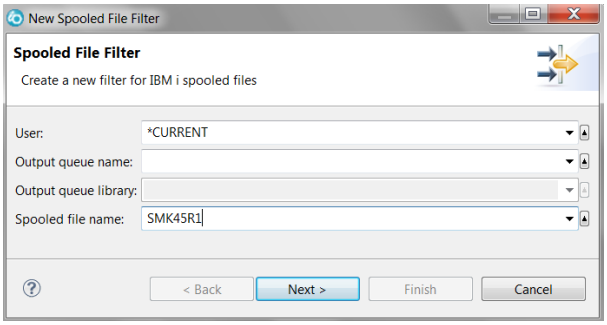


Figure 24: New Spooled File Filter 1

c. You can accept the name of the filter created by RDi and click **Finish**. This will create a filter entry in the **Spooled Files** list.

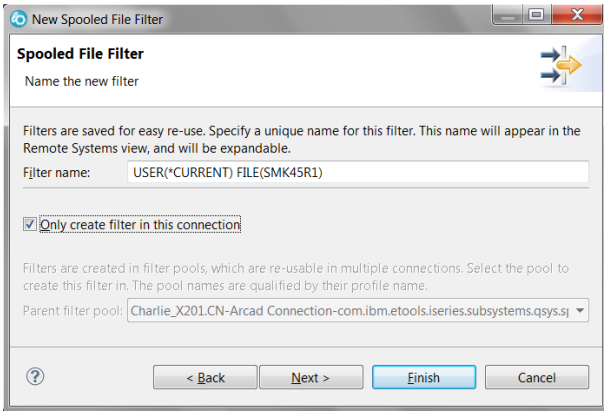


Figure 25: New Spooled File Filter 2

d. When you expand the list you will see *all spooled files related to this source member. Right-click on your selected file and RDi will allow you to view the spooled file.

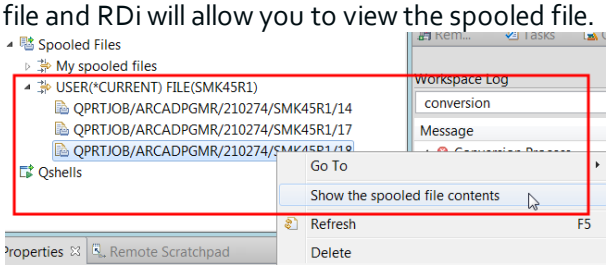
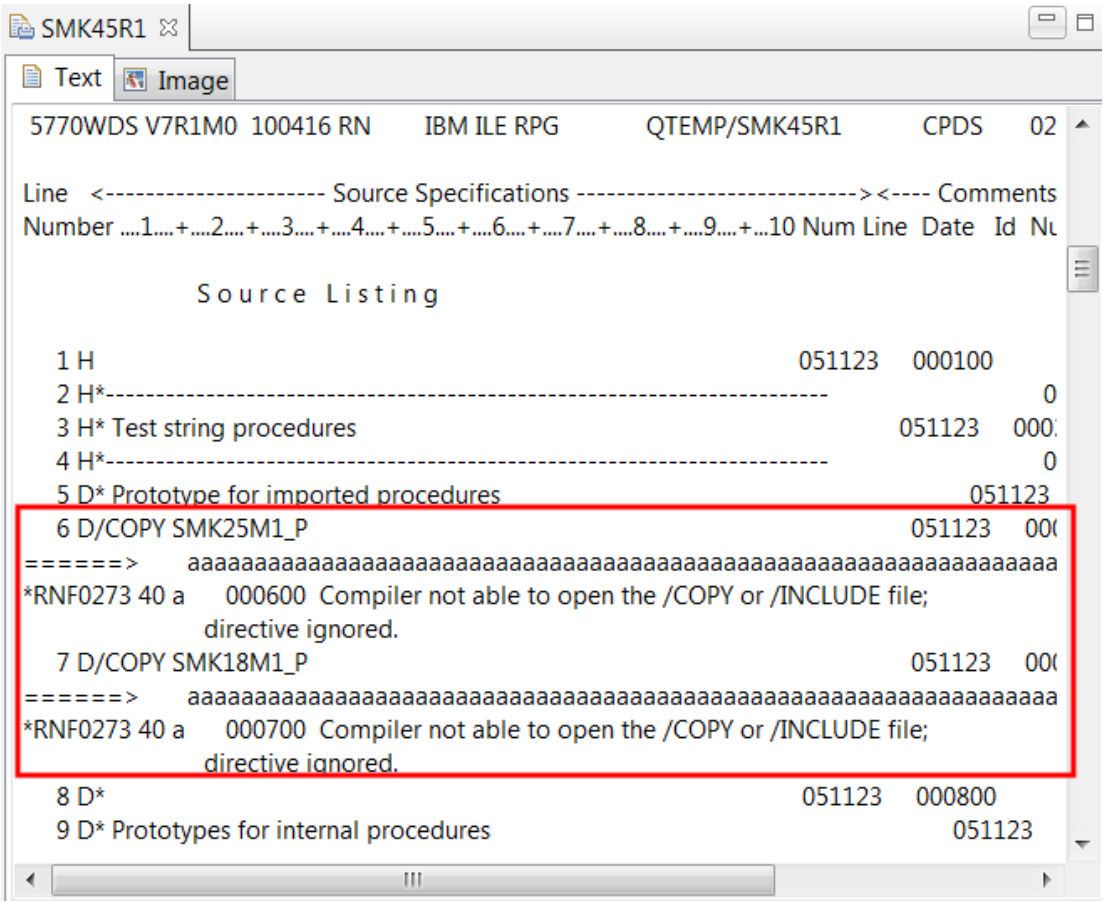


Figure 26: Show the spooled file contents

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com



```
5770WDS V7R1M0  100416 RN       IBM ILE RPG          QTEMP/SMK45R1          CPDS     02

Line   <--------------------- Source Specifications ---------------------------><---- Comments
Number ....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9....+...10 Num Line  Date   Id  N

                 S o u r c e   L i s t i n g

    1 H                                                                      051123     000100
    2 H*------------------------------------------------------------------------                  0
    3 H* Test string procedures                                                      051123     000
    4 H*------------------------------------------------------------------------                  0
    5 D* Prototype for imported procedures                                                    051123
    6 D/COPY SMK25M1_P                                                               051123     00
======>       aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
*RNF0273 40 a      000600  Compiler not able to open the /COPY or /INCLUDE file;
                  directive ignored.
    7 D/COPY SMK18M1_P                                                               051123     00
======>       aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
*RNF0273 40 a      000700  Compiler not able to open the /COPY or /INCLUDE file;
                  directive ignored.
    8 D*                                                                     051123     000800
    9 D* Prototypes for internal procedures                                                   051123
```

Figure 27: Spooled File Contents

Compatibility with the ARCAD Server

ARCAD RCP client applications, such as ARCAD-Verifier, ARCAD-Client or any other ARCAD, DOT or DROPS Eclipse-based studio, include a consistency version control. This control is based on the file *compliant.xml*.

You may have troubles connecting an RCP to the server after it is upgraded from one version to the next. The blocking message in red below may appear when you attempt to connect to the server.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

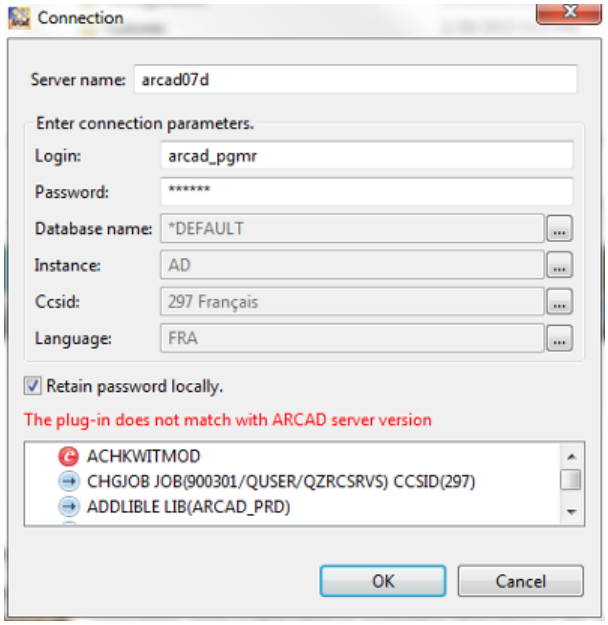


Figure 28: Server compatibility error

If you receive this error, install a new version of the RCP application, if a new version is available.

If no updated version of the RCP application has been released, and after carefully checking the compatibility aspects in the release notes, update the *compliant.xml* file as described below.

Updating the compliant.xml

> ***Warning!***
> This procedure should only be done in emergency situations. It is not recommended to use this as a permanent solution to server/studio compatibility problems.

The *compliant.xml* file is located in the following directory:

<installation>/plugins/com.arcadsoftware.core_<version>/compliant.xml.

Where **<installation>** is the RCP application installation directory and **<version>** is the highest version level in the plug-in directory.

Follow the subsequent steps to update the *compliant.xml* file.

Step 1 Check the existing version of the server using the following command (if on IBM i):

```
dspdtaara arcad_prd/arcversion
```

Step 2 Open the *compliant.xml* file for modification.

Step 3 Add a line based on one of the following examples:

- To be able to connect to a server running ARCAD version "aa.bb.cc":
  <version major="aa" minor="bb" release="cc" match="&PERFECT;"/>

- To be able to connect to a server running ARCAD aa.bb.whatever the release number:
  <version major="aa" minor="bb" release="XX" match="&PARTIAL;"/>

In this last case, a warning message will inform you that compatibility with the ARCAD version on the server is only PARTIAL.

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

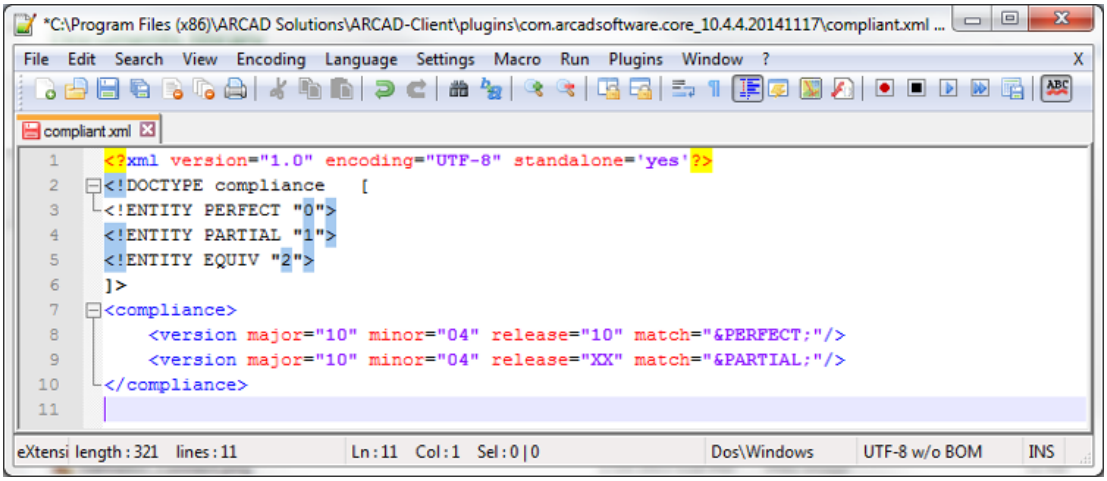


Figure 29: Editing the compliant.xml

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com

F.A.Q.

For more frequently asked questions, refer to our website.

Does the ARCAD-Transformer RPG Eclipse plug-in work in the 5250 interface?

The product is a standard Eclipse IDE plug-in. The product can be used from a 5250 interface, but in that case, you must use the `ARCAD_RPG/ACVTRPGFRE` command, which converts one member at a time.

It is not possible to perform mass-conversions using the 5250 interface unless other ARCAD products are installed.

To apply the license key using a 5250 session use the command `ARCAD_RPG/ALICCVTRPG`.

Does ARCAD-Transformer RPG handle conversion of MOVE or MOVEL?

The conversion of MOVE or MOVEL op codes are handled. See Move operations.

This operation code, widely used in traditional syntax, performs operations for which the behavior depends on the type of the variables and their length; all of the following cases are covered:

- Figurative constant in factor 2 (*Blank, *Zero, *Hival, *Loval, *ALL'o', *ALL'xxxx').
- With or without (p) as operation extender.
- Variable or fixed length field.
- Factor 2 field with a length less than the result field.
- Factor 2 field with a length greater than or equal to the result field.
- Assignment with numeric conversion alpha using %XLate, %Dec, %EditC, and possibly digital shifting by multiplying or dividing by 10, 100, 1000 etc...
- When factor 2 and/or the result field contain a Date/Time/TimeStamp field, conversion using %Date, %Time, %TimeStamp, or %Char, %Dec using the date/time format of the field.

If specified in positions 71-76, indicators are set after the converted instruction.

Does ARCAD-Transformer RPG handle OCL to CL?

No. The converter only converts ILE code to FREE!

ARCAD • Copyright © 2019 • All Rights reserved. • arcadsoftware.com